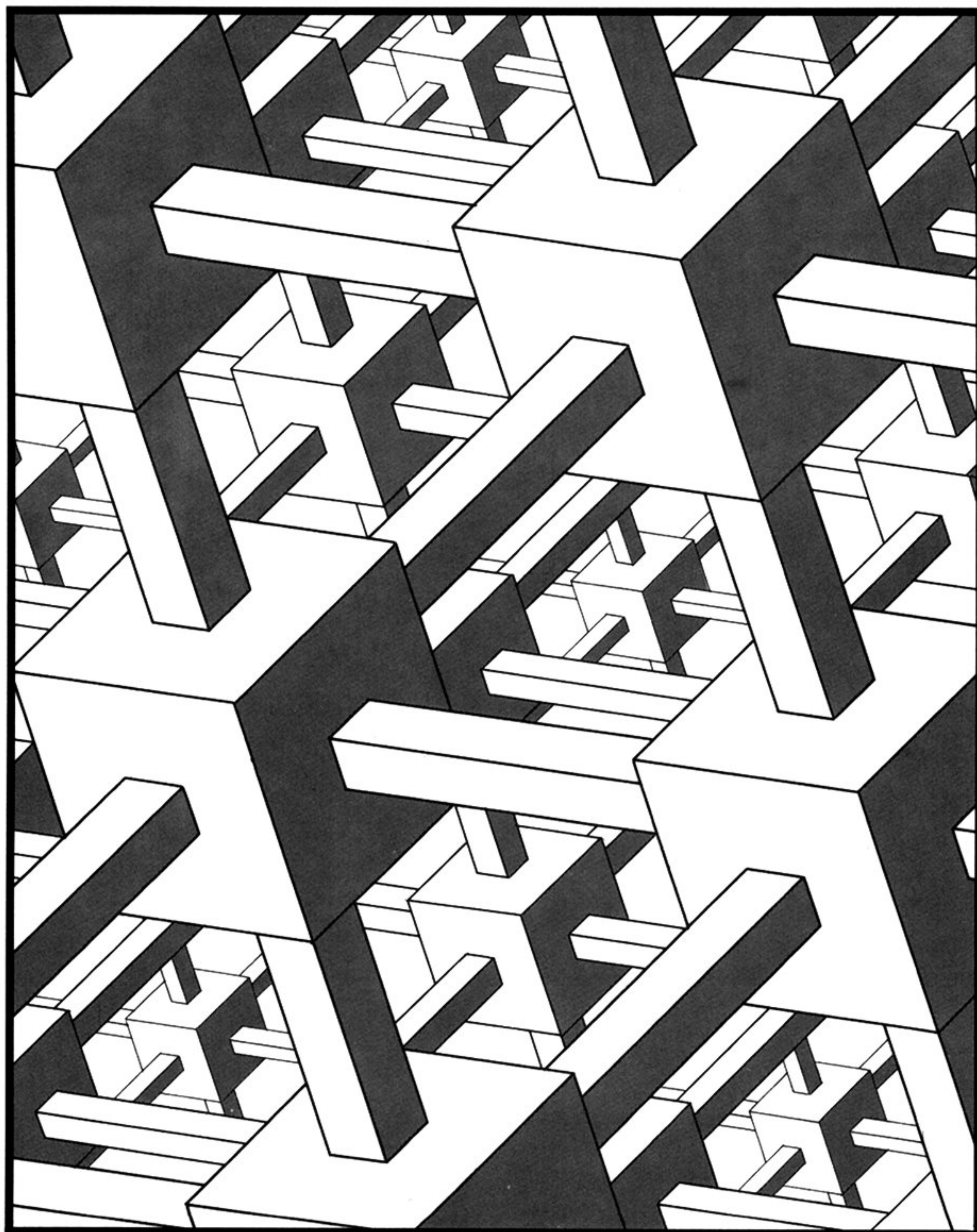
MSX-Datapack

エムエスエックス・データパック

Volume 2

ASCII

MSX-Datapack

エムエスエックス・データパック

Volume 2

ASCII

このマニュアルの読み方

このマニュアルの構成

このマニュアルは以下のように構成されています。

Volume 1

第1部　ハードウェア

第2部　システムソフトウェア

第3部　MSX-DOS

第4部　VDP

Volume 2

第5部　スロットとカートリッジ

スロットの切り換えなど、ソフトウェアから見たスロットやカートリッジについて解説します。

第6部　標準的な周辺装置のアクセス

MSXに標準で実装されている周辺装置(PSG、カセットインターフェイス、キーボードなど）の使用法を解説します。

第7部　オプションの周辺装置のアクセス

MSXの仕様ではオプションの周辺装置(RS-232C、MSX MODEM、MSX-MUSICなど）の使用法を解説します。

APPENDIX

BIOSやワークエリアなどの一覧表です。

このマニュアルの表記

1. MSX のバージョン

MSX1	MSX BASIC version 1.0 を搭載した MSX
MSX2	MSX BASIC version 2.0 を搭載した MSX
MSX2+	MSX BASIC version 3.0 を搭載した MSX
MSX	MSX1、MSX2、MSX2+の総称

2. キー

2つのキーが「+」で結ばれているときは、左側のキーを押しながら、右側のキーを押します。例えば、[CTRL] + [STOP] は、[CTRL] キーを押しながら [STOP] キーを押すことを示します。

3. BIOS エントリ、内部ルーチン

MSX2、MSX2+では BIOS が MAIN ROM と SUB ROM に分割されています。これを区別するため、以下のように表記します。

CHGET(009FH / MAIN)	MAIN ROM の 009FH 番地
REDCLK(01F5H / SUB)	SUB ROM の 01F5H 番地

4. ワークエリア、フック

【VALTYP(F663H, 1)】	F663H 番地の 1 バイトを使用
【BUF(F55EH, 258)】	F55EH 番地以降の 258 バイトを使用

5. その他、必要に応じて解説します。

目次

第7部　オプションの周辺機器

<MSX-Datapack　Vol.1　目次>

第１部　ハードウェア

第２部　システムソフトウェア

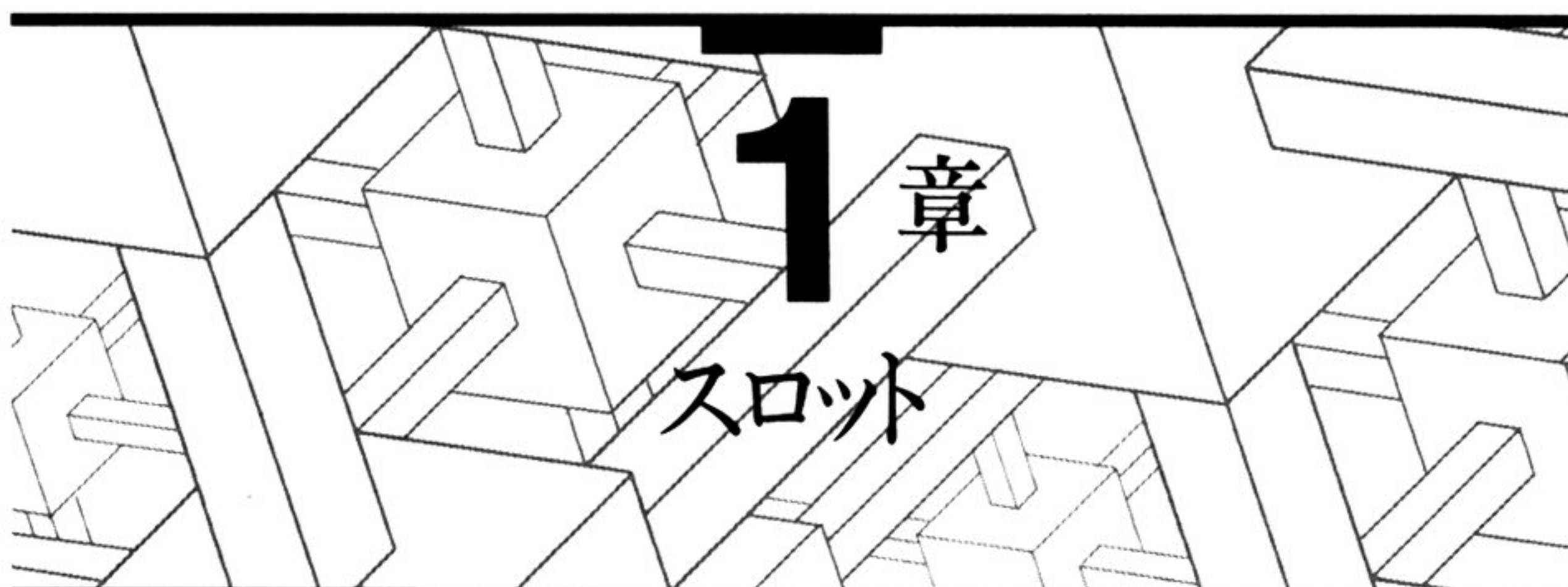
第3部　MSX-DOS

第4部　VDP

第5部
スロットとカートリッジ

MSXで使用されているCPU（Z80）は、通常64Kバイト（0000H～FFFFH）のアドレス空間しかアクセスできません。しかし、MSXは1M（メガ）バイトに相当する空間を自由にアクセスすることができます。これはMSXが「スロット」の概念を採用し、同一のアドレスに複数のメモリあるいはデバイスを割り当てる機能を備えているからです。

ここでは、このスロットの使用法、およびスロットを介してMSXにカートリッジソフトや新しいデバイスを接続するために必要となる情報を紹介します。

1章 スロット

スロットは大量のアドレス空間を確保するためのアドレス空間の切り換えをするもので、MSXのアドレスバスに接続されるメモリやデバイスは、すべてスロットを介して実装されています。それは、本体内にあるBASICのROMであろうとMSX-DOSモード時のRAMであろうと例外ではありません。カートリッジソフトを接続する場所も、1つのスロットです。ここでは、スロットに接続されたソフトウェアやデバイスの取り扱い方を説明します。

1.1 基本スロットと拡張スロット

スロットには基本スロットと拡張スロットの2種類があります。「基本スロット」とは、図5.1に示すようにCPUのアドレスバスに直結するスロットを指し、MSXの仕様では最大4個持つことができます。基本スロットは、スロット拡張ボックスを接続することにより（本体内で拡張されていることもある）、最大4個のスロットに拡張でき、このときのスロットを「拡張スロット」と呼びます。4個の基本スロットをそれぞれ4つの拡張スロットに拡張した場合、スロットの数は最大の16個となり、16スロット×64Kバイト=1Mバイトのアドレス空間をアクセスすることが可能です。

なお、拡張スロットにさらに拡張ボックスを差し込んだ場合はシステム自体が起動できなくなりますので、このようなことは行わないで下さい。MSX標準のカートリッジ用スロットは必ず基本スロットですが、各機種のオプションハードウェア専用コネクタは、拡張スロットに接続されていることがあります。

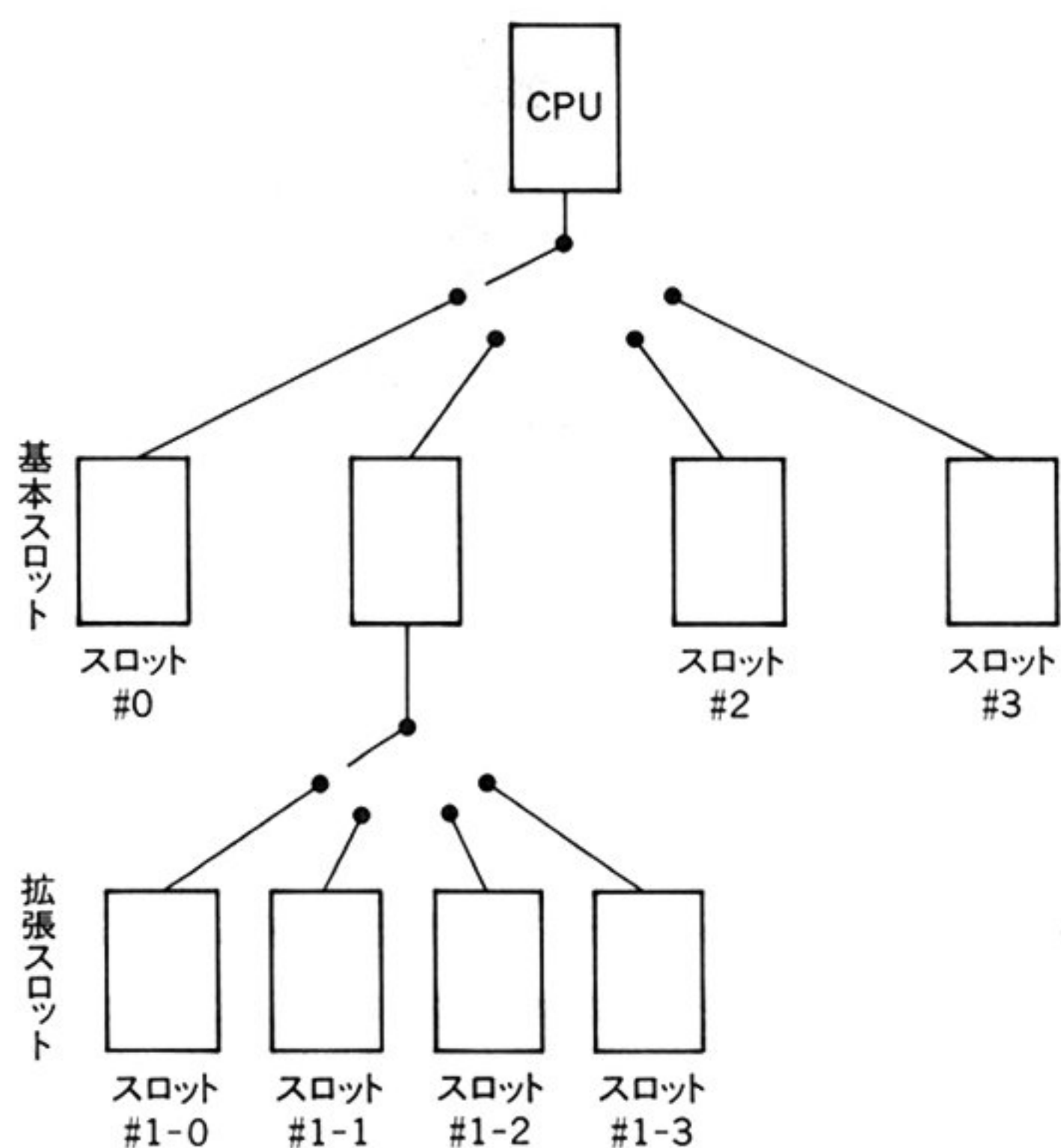



図 5.1 基本スロットと拡張スロット

各スロットは0000H〜FFFFHまでの64Kバイトのアドレス空間を持ちますが、MSXではそれを16Kバイトずつ4つに分け「ページ」として管理しています。CPUはページごとに任意のスロットを選択してアクセスでき、図5.2のように、いくつかのスロットから必要な部分だけを選んで組み合わせることが可能です。ただし、ある番号のページを異なる番号のページに割り当てることはできません。

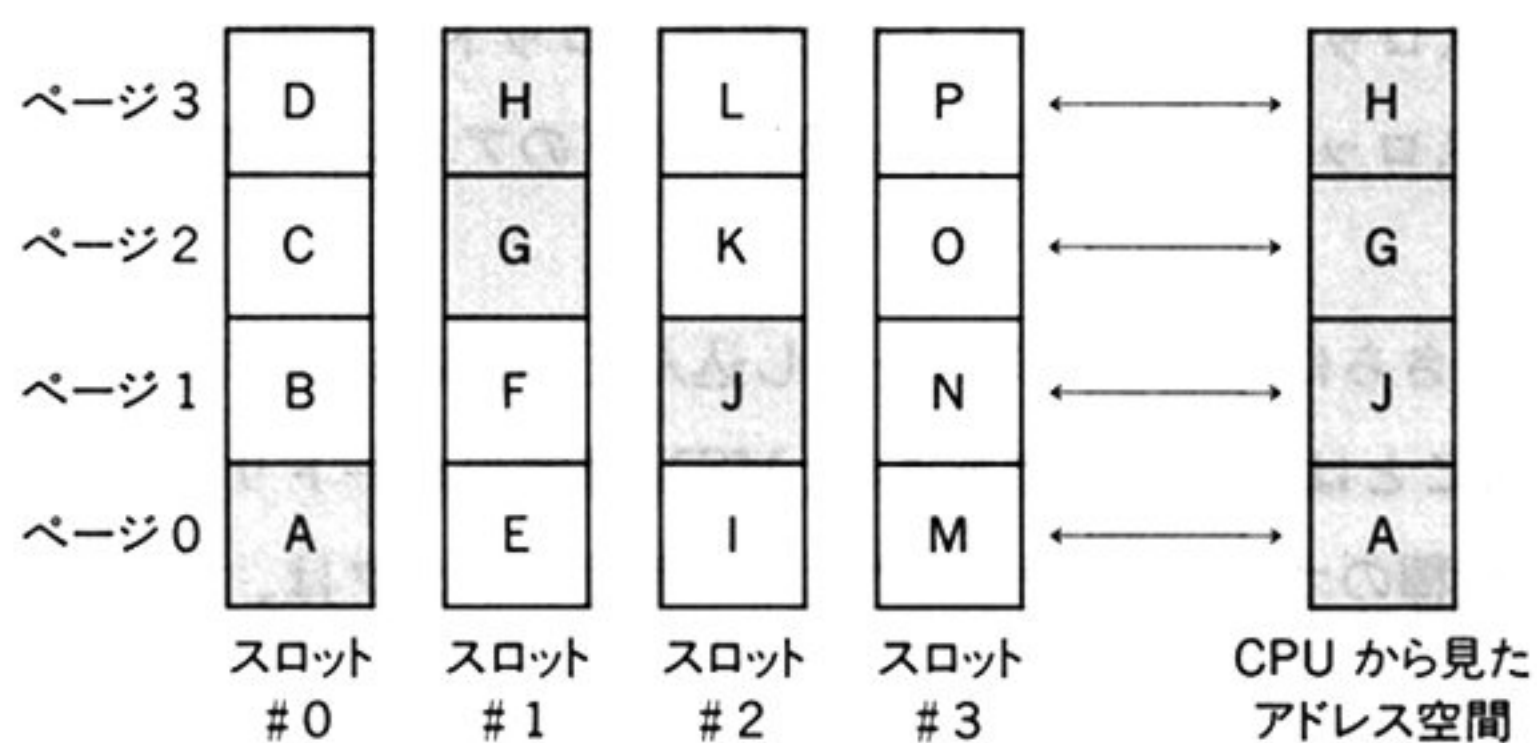



図 5.2 ページ選択の例

1.2 スロットの選択

スロットの選択方法は、基本スロットと拡張スロットでは異なります。基本スロットの場合はA8H番地のI/Oポートによって行い(図5.3)、拡張スロットの場合は実装された拡張カートリッジの「拡張スロット選択レジスタ (FFFFH)」によって行います (図5.4)。しかし、それらを直接変更することはたいへん危険ですから、スロットの切り換えはスロットの構造を理解した上で、BIOSのENASLT (0024H/MAIN) を呼び出して切り換えて下さい。他のスロットにあるプログラムを呼び出したい場合は、2章で説明するインタースロットコールを使用して下さい。

I/Oポート(A8H)

ページ0の基本スロット番号 (0〜3)
ページ1の基本スロット番号 (0〜3)
ページ2の基本スロット番号 (0〜3)
ページ3の基本スロット番号 (0〜3)

図5.3 基本スロットの選択

拡張スロット選択レジスタ(0FFFFH)

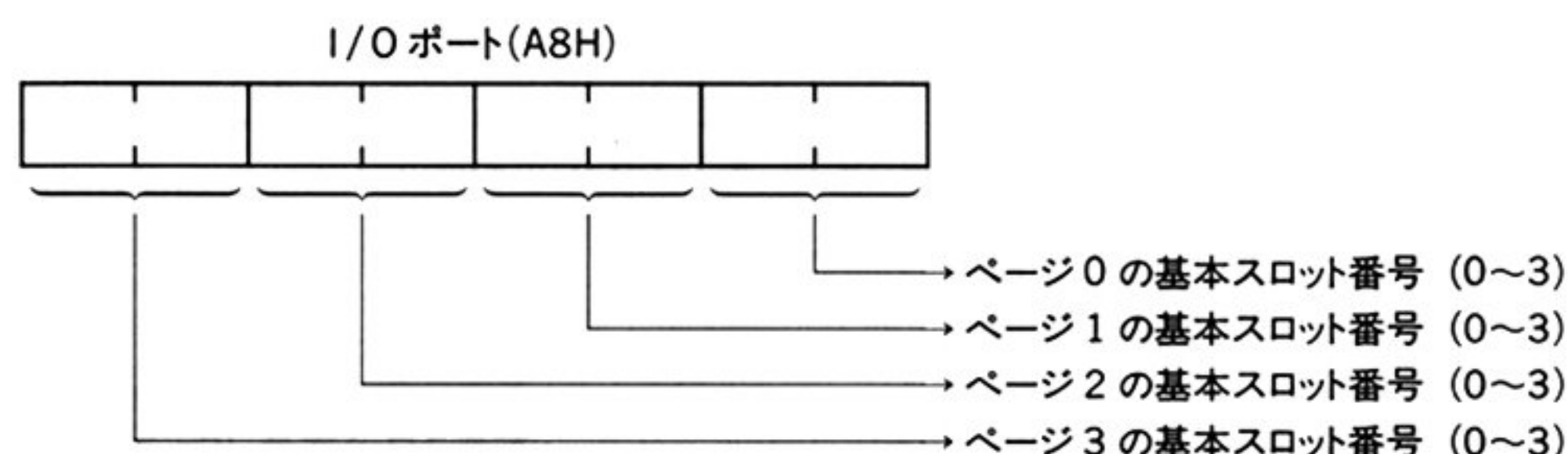
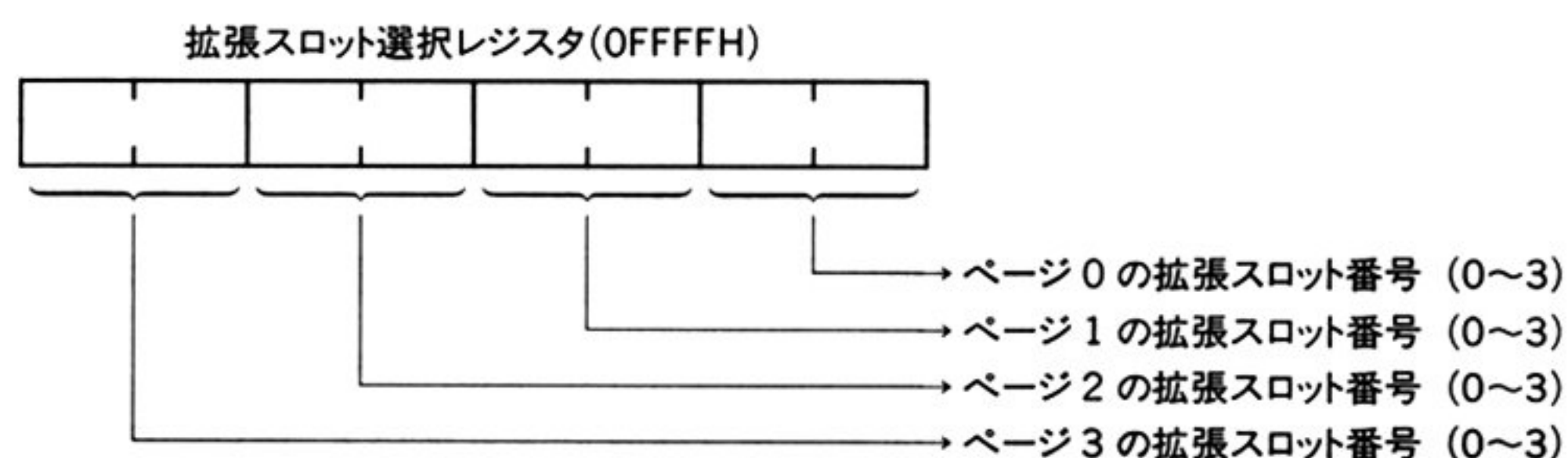
ページ0の拡張スロット番号 (0〜3)
ページ1の拡張スロット番号 (0〜3)
ページ2の拡張スロット番号 (0〜3)
ページ3の拡張スロット番号 (0〜3)

注) このスロットが基本スロットか、拡張スロットかが判るように、拡張スロットの場合は値を書き込んだ後に読み出すと、書き込んだ値が反転したものが値として読み出されます。
このレジスタは同一基本スロット内であれば、どの拡張スロットでも同じ値です。

図5.4 拡張スロットの選択

どこのスロットにMAIN ROMやRAMが実装されているか、またカートリッジ用のスロットが何番のスロットであるかは機種によって異なります。もし手持ちのMSXがどのようにスロットを使用しているかを知りたい人はマニュアルなどで調べて下さい。しかし、MSXはどのスロットに何があろうと正常な動作ができるように仕様が決められていますから、その仕様に準じている限り、通常はスロットの使用状況を気にする必要はありません。

しかし、場合によっては特定のソフトウェアがどこのスロット上に置かれているのかを知る必要が生じることもあります。例えば、BASICのMAIN ROMは基本スロット#0、または基本スロット#0を拡張した拡張スロット#0-0に置くという仕様ですが、MSX1にMSX2の機能を追加するMSX2バージョンアップアダプタでは、MAIN ROMがスロット#0やスロット#0-0以外の上に置かれることになります。また、MSX2のSUB ROMが入っているスロットは機種によってまちまちで、このような場合、下記のワークエリアを参照することにより、BASICインタープリタのROMが置かれているスロットを知ることが可能です。スロット情報は図5.5のフォーマットで得られます。DOSからBIOSを呼び出すときも、この方法を使用してMAIN ROMのスロットを調べて下さい。

【EXPTBL(0FCC1H,1)】 MAIN ROMのスロット

【EXBRSA(FAF8H,1)】 SUB ROMのスロット (MSX1では0)

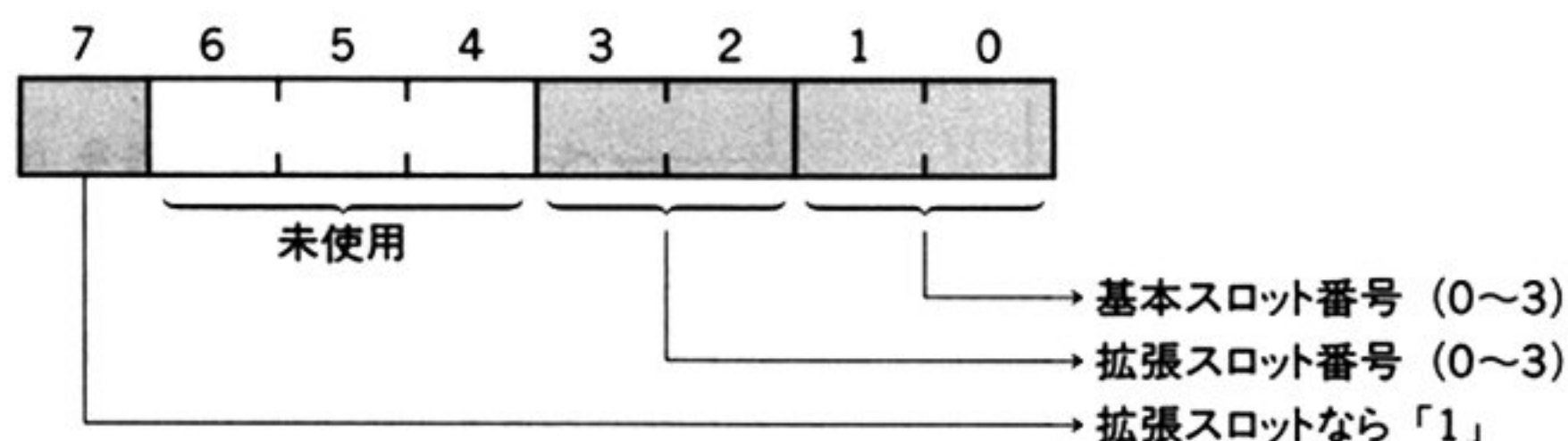



図5.5 スロットのフォーマット

また、あるルーチンがページ1とページ2(4000H〜BFFFH)にまたがっている場合、このルーチンの中で1ページから2ページへ分岐が起こるときには、ページ2にページ1と同じスロットが選択されていなければなりません。そのためには、ページ1でまず自分のいるスロットを調べ、ページ2をそのスロットに切り換えるという作業が必要です。現在自分がどのスロットにいるのか、という情報を知るためには、添付のフロッピーディスクに入っている「WHEREAMI.MAC」を使用します。

2章 インタースロットコール

前述のように、MSX ではプログラムが異なったスロットに分かれているため、現在選択されているスロット上にはないプログラムが必要になることもあります。これは主に以下のような場合が考えられます。

1. MSX-DOS の環境から、MAIN ROM にある BIOS を呼び出す。
2. BASIC の環境から、SUB ROM にある BIOS を呼び出す（MSX2 以降）。
3. カートリッジソフトから、MAIN ROM あるいは SUB ROM の BIOS を呼び出す。

これらの作業を行う際、スロット切り換えが簡単かつ安全に行えるように、インタースロットコールという一群の BIOS ルーチンが用意されていて、どのスロットに存在するルーチンでも呼び出せるようになっています。この章では、このインタースロットコールの使用法を説明します。

2.1 インタースロットコールの動作

例えば、MSX-DOS から MAIN ROM 上の BIOS を呼び出す場合、スロットの状態の遷移は以下に示すとおりです。

1. 初めは MSX-DOS 環境なので、64K のアドレス空間すべてに RAM が選択されている。このままでは BASIC-ROM をアクセスできない（図 5.6-1）。
2. ROM の BIOS を呼び出すために、ページ 0 を BASIC の MAIN ROM に切り換え、アクセス可能な状態にする。そして BIOS を呼び出す（図 5.6-2）。
3. BIOS の処理が終わったら再び元の状態に戻し、初めに呼ばれたアドレスにリターンする。

1. MSX-DOS を使用している状態

	MSX-DOS	BASIC-ROM
ページ3	RAM	なし
ページ2	RAM	なし
ページ1	RAM	ROM
ページ0	RAM	ROM
	スロット 3	スロット 0

2. BASIC-ROM の BIOS をコールしている状態

	MSX-DOS	BASIC-ROM
ページ3	RAM	なし
ページ2	RAM	なし
ページ1	RAM	ROM
ページ0	RAM	ROM BIOS が呼び出される
	スロット 3	スロット 0

（網掛け）が CPU 空間にイネーブルされる

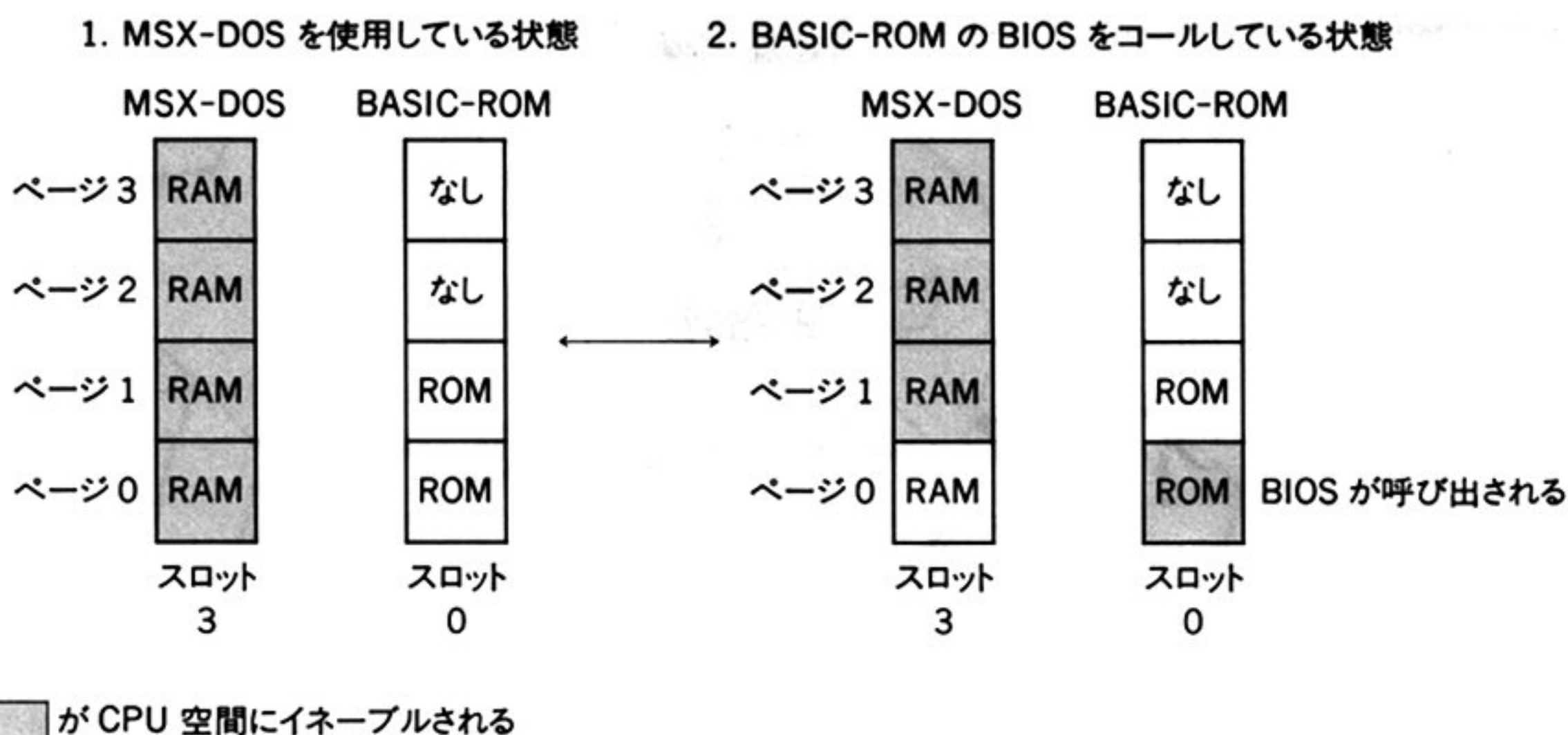



図 5.6 インタースロットコールのされ方

このとき、MSX-DOS 側のプログラムが、ページ0以外に存在するならば簡単なのですが、呼び出す側のプログラムが呼び出される側の BIOS と同じページ0に存在しているときは、非常に複雑になります。呼び出す方のプログラムがページ0を切り換えたとたんに自分自身がいなくなって暴走してしまう、ということのないように配慮しなければいけません。

インタースロットコールでは、いったんページ3へ分岐してから実際のスロット切り換えを行うという方法で、この問題を解決しています。

このような複雑な状況をインタースロットコールはサポートするので、ユーザーは簡単に他のスロットのプログラムを呼び出すことができます。

2.2 インタースロットコールの使用法

インタースロットコールは、以下に述べるいくつかの方法で行うことができます。これらは BIOS として MAIN ROM 内に含まれているものですが、その中のいくつかは MSX-DOS の環境下においてもまったく同一のものが用意され、MSX-DOS 使用時のインタースロットコールを可能としています（「2.2.2 MSX-DOS のインタースロットコール」参照）。

2.2.1 BIOSのインタースロットコールルーチン

RDSLT(000CH / MAIN)

機　能

指定スロットの指定アドレスから値を読み出します。

コール手順

A　　スロット指定（図5.5と同じフォーマット）
HL　　読み出すアドレス

戻り値

A　　読み出した値

変更レジスタ

AF、BC、DE

解　説

指定したスロットの指定したアドレスの内容を読み出し、Aレジスタに格納します。スロットの指定はAレジスタによって図5.5の形式で行います。このとき、目的のスロットが基本スロットならば、上位6ビットはすべて「0」に設定し、下位2ビットでスロット#0〜#3を決めます。もし拡張スロットを指定するならば、同様にビット0とビット1で基本スロットを指定し、その基本スロットに接続されるどの拡張スロットかということをビット2とビット3で指定、さらにビット7を「1」にします。

WRSLT(0014H / MAIN)

機　能

指定スロットの指定アドレスに値を書き込みます。

コール手順

A　　スロット指定（図5.5と同じフォーマット）
HL　　書き込むアドレス

E　　書き込む値

戻り値

なし

変更レジスタ

AF、BC、D

解　説

Aレジスタで指定したスロット（指定の形式は図5.5と同じ）のHLレジスタで指定したアドレスに、Eレジスタの値を書き込みます。

CALSLT（001CH / MAIN）

機　能

指定したスロットの指定アドレスをコールします。

コール手順

IY	の上位8ビット	スロット（図5.5と同じフォーマット）
IX		コールするアドレス

戻り値

呼び出し先プログラムの実行結果により異なる

変更レジスタ

呼び出し先プログラムの実行結果により異なる

解　説

IYレジスタの上位8ビットで指定したスロット（指定の形式は図5.5と同じ）の、IXレジスタで指定するアドレスに存在するルーチンをコールします。

ENASLT(0024H / MAIN)

機　能

スロットを切り換えます。

コール手順

A　　スロット（図5.5と同じフォーマット）
HL　　上位2ビットでスロット切り換えを行うページを指定する

戻り値

なし

変更レジスタ

すべて

解　説

HLレジスタの上位2ビットで指定したページを、Aレジスタで指定したスロットに切り換えます。

CALLF(0030H / MAIN)

機　能

指定スロットの指定アドレスをコールします。

コール手順

インラインパラメータ形式でスロットとアドレスを指定する

戻り値

呼び出し先のプログラムの実行結果により異なる

変更レジスタ

呼び出し先のプログラムの実行結果により異なる

解　説

指定したスロットの指定したアドレスをコールしますが、前述のCALSLTと異なり、ス

ロットおよびアドレスの指定は次に示すように、インラインパラメータ形式で行います。すなわち、このCALLFを呼び出す命令の直後にスロットを指定する値1バイト(RDSLTと同じフォーマット)を置き、その次にアドレスを指定する数値2バイトを置くという形でパラメータを渡します。「CALL 0030H」の代わりに「RST 30H」というRST(リスタート)命令を使ってもかまいません。その場合、4バイトでインタースロットコールが実現可能になります。

リスト5.1　インタースロットコールの実行例

```
rst     30h         ; interslot call
db      00000000b   ; select slot # 0
dw      006ch       ; call address = 006ch
```

RSLREG(0138H / MAIN)

機　能

基本スロット選択レジスタの内容を読み出します。

コール手順

なし

戻り値

A　　読み込んだ値（図5.5と同じフォーマット）

変更レジスタ

A

解　説

基本スロット選択レジスタの内容を読み出し、Aレジスタに入れます。

WSLREG(013BH / MAIN)

機　能

基本スロット選択レジスタへ値を書き込みます。

コール手順

A　書き込む値（図5.5と同じフォーマット）

戻り値

なし

変更レジスタ

なし

解　説

基本スロット選択レジスタにAレジスタの値を書き込み、スロットを選択します。

SUBROM(015CH / MAIN)

機　能

SUB ROMの指定したアドレスをコールします。

コール手順

IX　コールするアドレス、PUSH IX（「A.1.2 SUB ROM」参照）

戻り値

呼び出し先のプログラムの実行結果により異なる

変更レジスタ

裏レジスタとIX、IYレジスタは保存される

解　説

BASICのSUB ROMを呼び出すためのルーチンです。SUB ROMの存在するスロットはBIOSが自動的に調べます。普通は、次のEXTROMを使います。

EXTROM(015FH / MAIN)

機　能

SUB ROMの指定したアドレスをコールします。

コール手順

IX　　コールするアドレス

戻り値

呼び出し先のプログラムの実行結果により異なる

変更レジスタ

裏レジスタとIYレジスタは保存される

解　説

BASICのSUB ROMを呼び出すためのルーチンです。上記のSUB ROMとこのEXTROMとは、IXレジスタの値をpushするか否かという点だけが異なります。

2.2.2 MSX-DOSのインタースロットコール

MSX-DOSでは、以下に示す5種類のインタースロットコールが用意され、MSX-DOSジャンプベクタにそのエントリアドレスが定義してあります。これらはBIOS内の同名のルーチンとまったく同じものですから、機能や使用法に関しては、前述のBIOSをご覧下さい。

なお、MSX-DOSからSUB ROM内のルーチンを呼び出す場合は、これらのルーチンは使用せず、「2.2.3 MSX-DOSからのSUB ROMコール」の方法を使います。

RDSLT(000CH)	指定スロットの指定アドレスから値を読み出す
WRSLT(0014H)	指定スロットの指定アドレスに値を書き込む
CALSLT(001CH)	指定スロットの指定アドレスを呼び出す
ENASLT(0024H)	指定スロットを使用可能な状態にする
CALLF(0030H)	指定スロットの指定アドレスを呼び出す

リスト 5.2　MSX-DOS から BIOS を呼び出す

```
calslt   equ    001ch            ; inter slot call
exptbl   equ    0fcc1h           ; slot address of MAIN ROM

         ld     iy,(exptbl-1)    ; load slot address of MAIN ROM
                                 ; in high byte of IY
         ld      ix,address of the BIOS jump table
         call   calslt
```

2.2.3 MSX-DOS からの SUB ROM コール

MSX-DOS の環境から SUB ROM を呼ぶ場合は、以下のような特別な配慮が必要です

1. CALSUB のエントリ条件は、IX にアドレスを入れるだけです。
2. スタックはページ 0 以外でなければなりません。
3. 割り込み禁止は必要ありません。
4. 終了後、NMI フックの解除は必要ありません。

ここで問題となるのは、EXTROM に必要な IX レジスタの値が渡せないことす。したがって、BIOS ROM が呼び出された後に IX がセットされ、EXTROM にジャンプする必要があります。

具体的には、スタック上に次のようなルーチンを用意し、NMI のフックからジャンプさせるようにしておきます。次に、BIOS の NMI エントリ（フックは用意されているが使用されない）をインタースロットコールします。リスト 5.3 を参照して下さい。

リスト 5.3 DOS からの SUB ROM コール

```
;         +0    inc     sp
;         +1    inc     sp
;         +2    ld      ix,<sub-ROM entry>
;         +6    nop
;         +7    jp      extrom
calslt    equ     001ch
extrom    equ     015fh
nmi       equ     0066h          ; Non-maskable interrupt
h.nmi     equ     0fdd6h         ; Hook for NMI
exptbl    equ     0fcc1h
;
_calsub::
          exx                    ; Save argument registers over setup
          ex      af,af'
          ld      hl,extrom
          push    hl
          ld      hl,0c300h      ; jp xxxx, nop
          push    hl
          push    ix             ; SUB ROM entry
          ld      hl,021ddh      ; ld ix,xxxx
          push    hl
          ld      hl,03333h      ; inc sp, inc sp
          push    hl
          ld      hl,0
          add     hl,sp
          ld      a,0c3h
          ld      (h.nmi),a
          ld      (h.nmi+1),hl
          ex      af,af'         ; Restore registers
          exx
;
          ld      ix,nmi
          ld      iy,(exptbl-1)
          call    calslt
          ei
;
          ld      hl,10          ; Throw away the interface routine
          add     hl,sp
          ld      sp,hl
          ret

          end
```

2.3 スロットの状態を知るためのワークエリア

スロットに関するワークエリアには、以下のものがあります。

【EXBRSA(0FAF8H,1)】　SUB ROMのスロットアドレス

7	6	5	4	3	2	1	0
	未使用						

- bit 0〜1 → 基本スロット番号（0〜3）
- bit 2〜3 → 拡張スロット番号（0〜3）
- bit 7 → 拡張スロットなら「1」

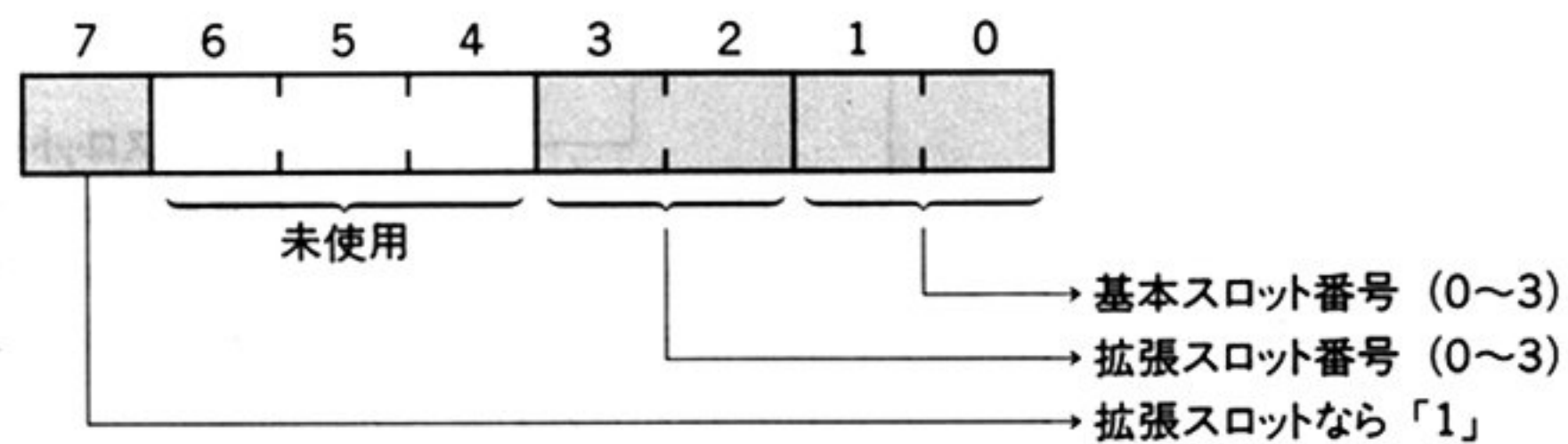



図5.7　SUB ROMのスロットアドレス

【EXPTBL(0FCC1H,4)】　基本スロットの拡張の有無

	7	6	5	4	3	2	1	0	
【0FCC1H】									MAIN ROMのスロット
【0FCC2H】		未使用							スロット#1
【0FCC3H】		未使用							スロット#2
【0FCC4H】		未使用							スロット#3

bit 7:
- 0　スロットは拡張されていない
- 1　スロットは拡張されている

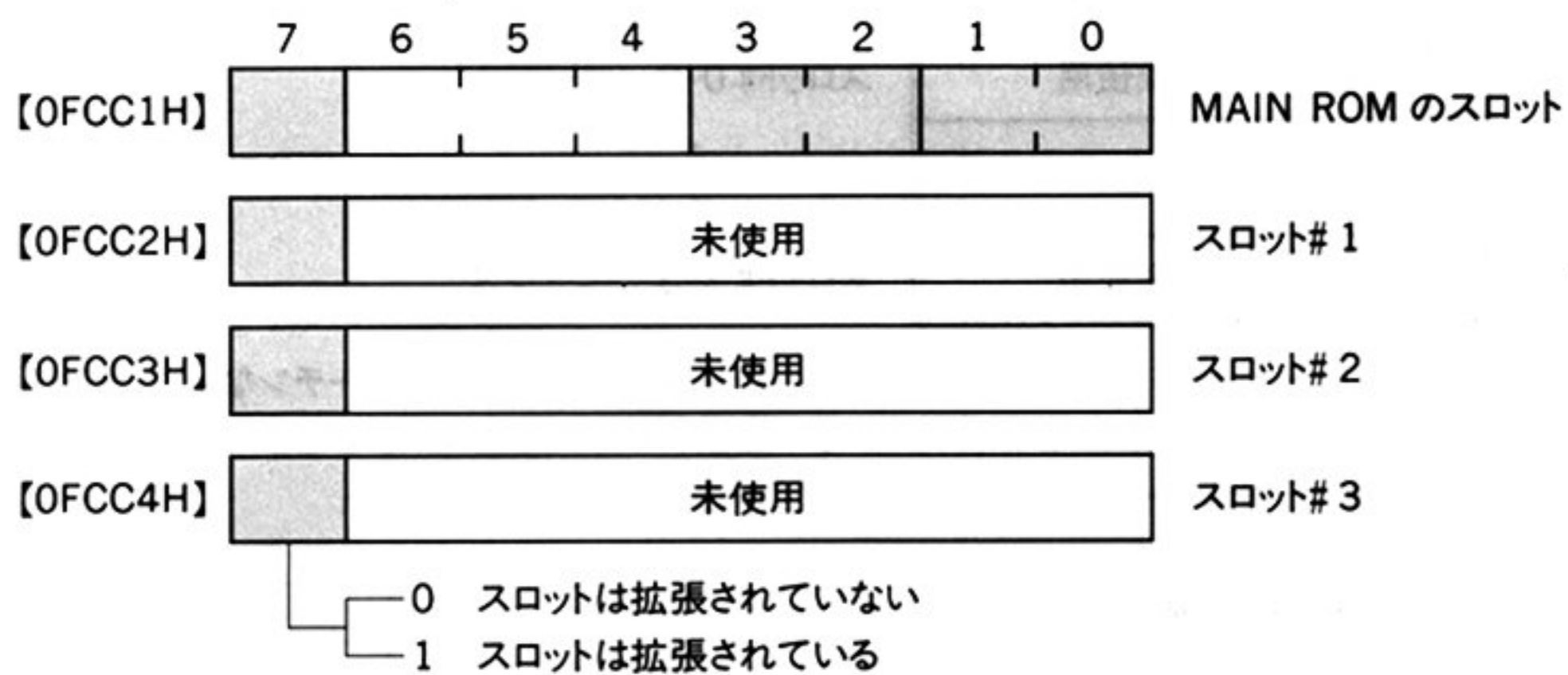



図5.8　基本スロットの選択

【SLTTBL(0FCC5H,4)】　拡張スロット選択レジスタ値の保存エリア

7 6 5 4 3 2 1 0

【0FCC5H】 スロット#0の拡張スロット選択レジスタの値
【0FCC6H】 スロット#1の拡張スロット選択レジスタの値
【0FCC7H】 スロット#2の拡張スロット選択レジスタの値
【0FCC8H】 スロット#3の拡張スロット選択レジスタの値

→ページ0の拡張スロット番号
→ページ1の拡張スロット番号
→ページ2の拡張スロット番号
→ページ3の拡張スロット番号

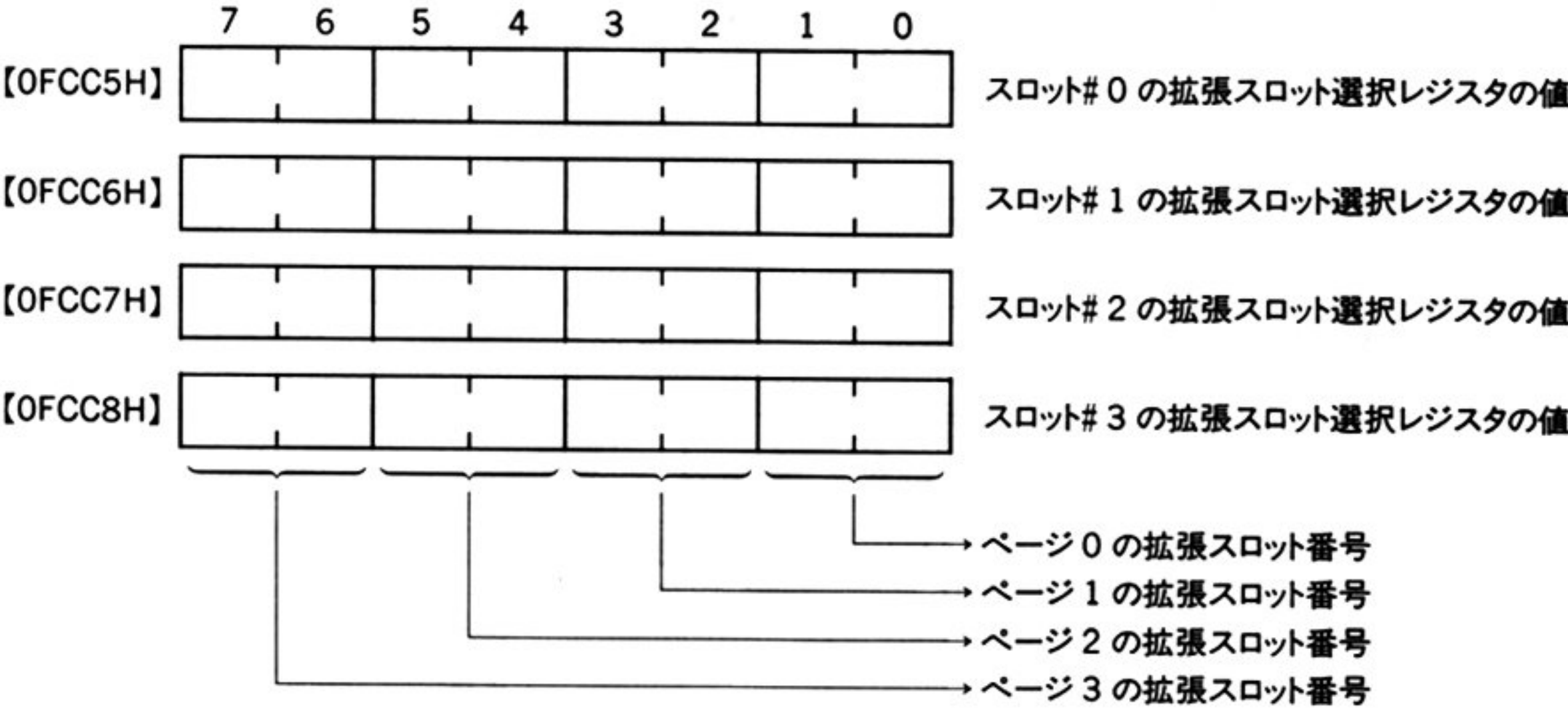



図5.9　拡張スロットの選択

【SLTATR(0FCC9H,64)】　各スロット、ページにおけるアプリケーションの有無

【0FCC9H】 未使用 スロット#0-0、ページ0
【0FCCAH】 未使用 スロット#0-0、ページ1
⋮
【0FD08H】 未使用 スロット#3-3、ページ3

→0　拡張ステートメント処理ルーチンなし　1　あり
→0　拡張デバイス処理ルーチンなし　1　あり
→0　BASICテキストなし　1　あり

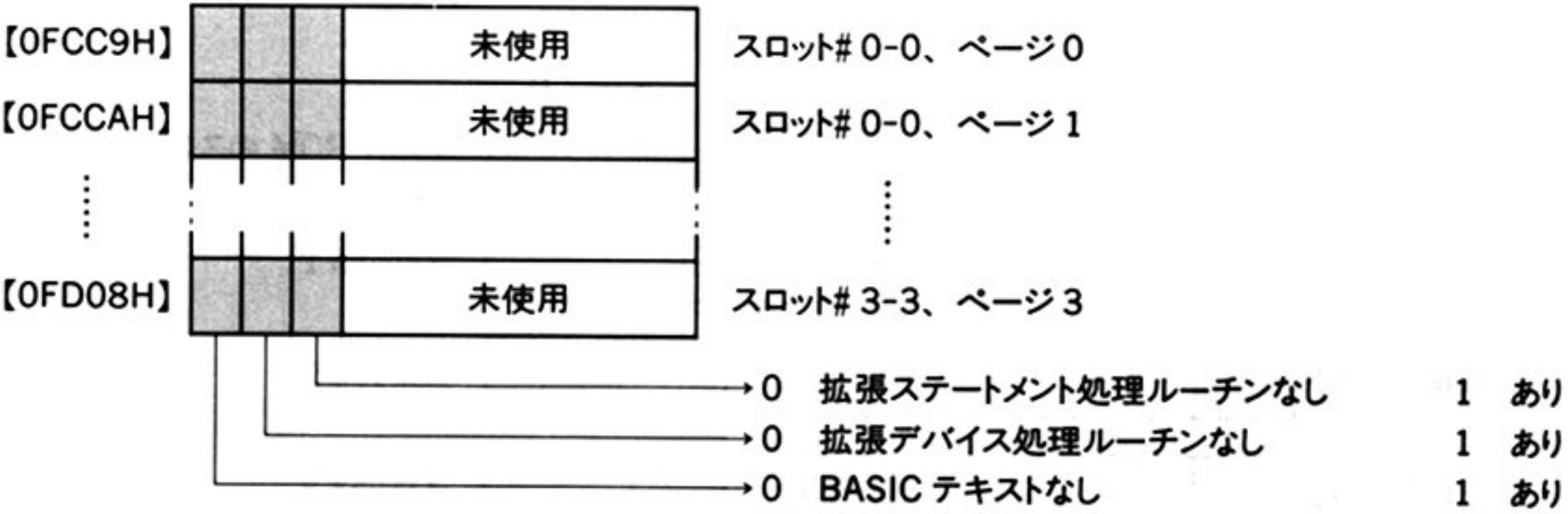



図5.10　アプリケーションの有無

【SLTWRK(0FD09H,128)】　アプリケーション用ワークエリア

【0FD09H】 スロット#0-0、ページ0のワークエリア
【0FD0BH】 スロット#0-0、ページ1のワークエリア
⋮
【0FD85H】 スロット#3-3、ページ2のワークエリア
【0FD87H】 スロット#3-3、ページ3のワークエリア

1ページにつき2バイト

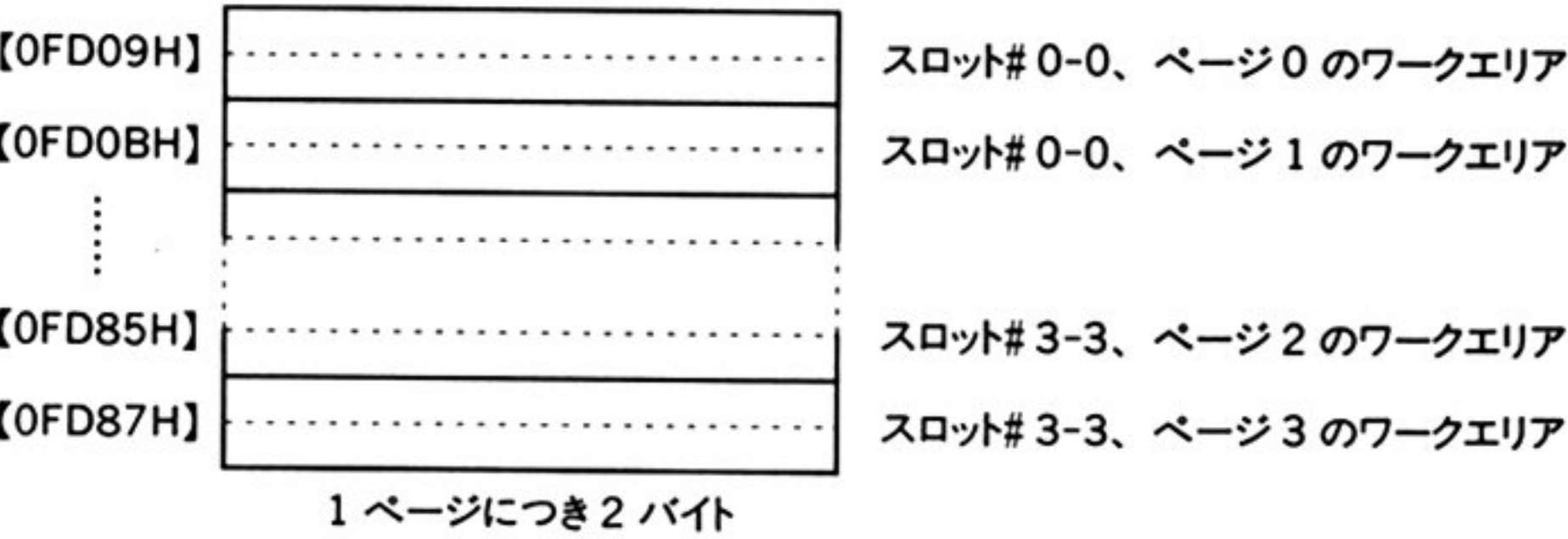



図5.11　アプリケーション用ワークエリア

3章 カートリッジソフトの作成法

MSXには最低1つのスロットが表に出ており、ここに接続されるハードウェアを「カートリッジ」といいます。カートリッジには、

1. アプリケーションプログラムやゲームなどのROMカートリッジ
2. フロッピーディスクインターフェイスやRS-232Cインターフェイスなどの入出力装置カートリッジ
3. RAMを増やすための増設RAMカートリッジ
4. スロットを増やすためのスロット拡張器

などいろいろと接続でき、これによりMSXは機能の拡張を簡単に行うことができます。また、BASICやマシン語プログラムをROM化（カートリッジ化）することも簡単です。この章では、ソフトウェアをカートリッジ化する方法を説明します。

3.1 カートリッジヘッダ

MSXのカートリッジは、16バイトの共通ヘッダを持っており、システムがリセットされると、このヘッダに記述された情報によりカートリッジの初期設定が行われます。BASICやマシン語のプログラムをROM化したカートリッジの場合は、このヘッダに書き込む情報によってオートスタートさせることも可能です。カートリッジヘッダの構成を図5.12に示します。

4000H または 8000H

オフセット	内容
+0000H	ID
+0002H	INIT
+0004H	STATEMENT
+0006H	DEVICE
+0008H	TEXT
+000AH	システム予約*
+0010H	

*システム予約領域は 00H で埋めて下さい。

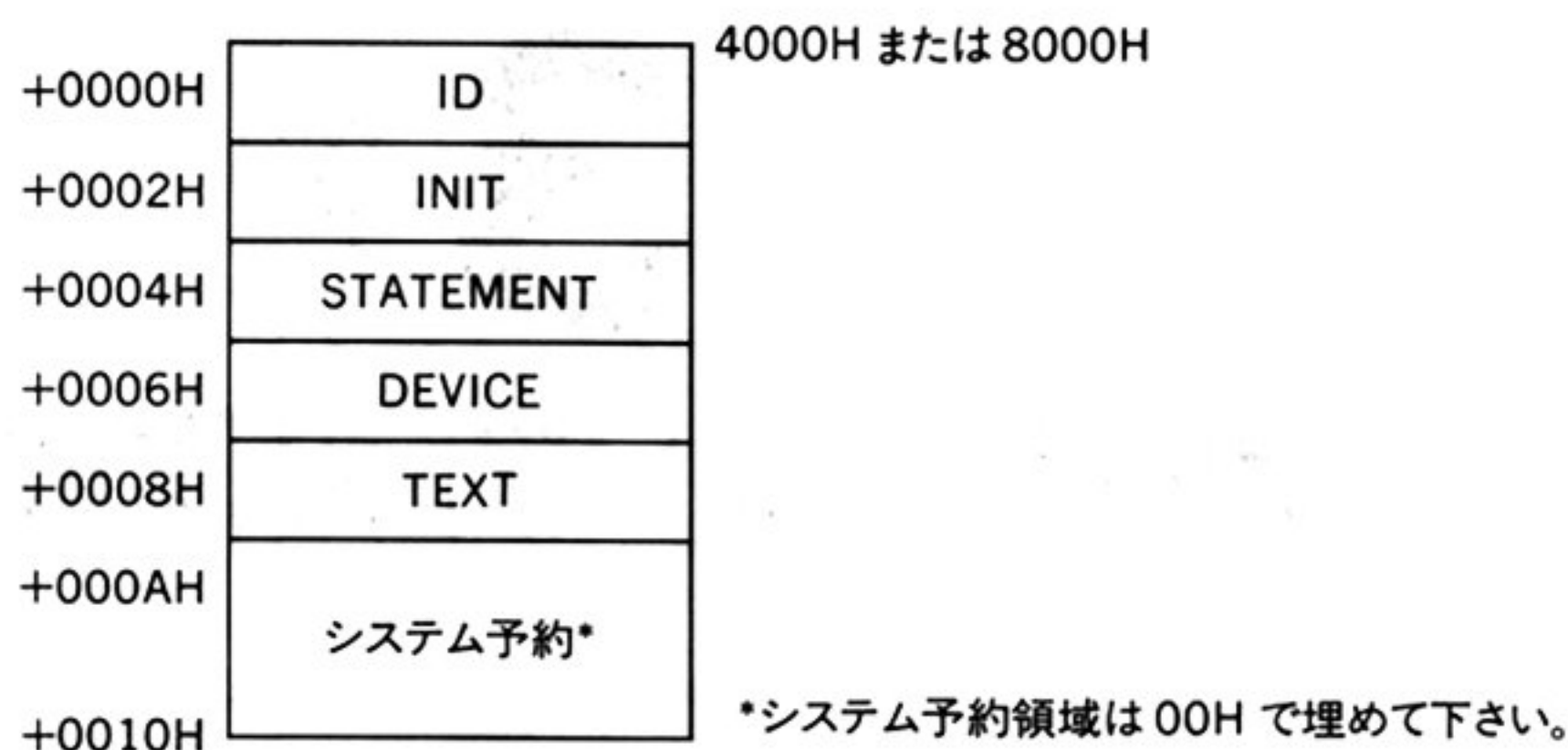



図 5.12　プログラムカートリッジのヘッダ

■ ID

ROM カートリッジであれば、この 2 バイトに「AB」(41H、42H) というコードを入れておきます。ちなみに SUB ROM の ID は「CD」となっています。

■ INIT

この 2 バイトには、カートリッジがワークエリアや I/O などの初期化を行うのであれば、その初期化ルーチンのアドレスを書き込み、行わなければ 0000H としておきます。初期化ルーチンの中でワークエリアの確保など必要な処理を行ったら、「RET」で終了させて下さい。このとき SP 以外のレジスタは内容を破壊してもかまいません。なお、ゲームのようにカートリッジ内でループしていればよいマシン語プログラムの場合は、ここからそのまま目的のプログラムを実行することが可能です。

■ STATEMENT

この 2 バイトには、カートリッジが CALL 文の拡張を行うのであれば、ステートメント拡張ルーチンのアドレスを書き込み、行わなければ 0000H としておきます。CALL 文の拡張を行う場合、ステートメント拡張ルーチンは 4000H～7FFFH になければいけません。

CALL 命令は以下の書式で記述されます。

CALL＜拡張ステートメント名＞[(＜引数＞[,＜引数＞･･･])]

拡張ステートメント名は 15 文字以下でなければいけません。CALL の省略として「_」(アンダースコア) も使用できます。

BASIC インタープリタは CALL 文を見つけると、ワークエリアの【PROCNM(0FD89H,16)】に拡張ステートメント名を入れ、ヘッダの STATEMENT の内容が 0 以外のカートリッジにス

ロット番号の小さい方から順に制御を渡してきます。このときHLレジスタは拡張ステートメント名の次のテキストアドレスを指しています（図5.13-1）

拡張ステートメントの作成については、「第2部 5.5 拡張ステートメント」も参照して下さい。

1. インタープリタが拡張ステートメント処理ルーチンを呼ぶときの入力設定

```
CALL ABCDE (0, 0, 0):A = 0
           ↑
           HL
```

CY フラグ= 1

拡張ステートメント名の終わり

PROCNM	A	B	C	D	E	00H

2. 拡張ステートメントの処理を行わなかった場合の出力設定

```
CALL ABCDE (0, 0, 0):A = 0
           ↑
           HL
```

CY フラグ= 1

3. 拡張ステートメントの処理を行った場合の出力設定

```
CALL ABCDE (0, 0, 0):A = 0
                    ↑
                    HL
```

CY フラグ= 0

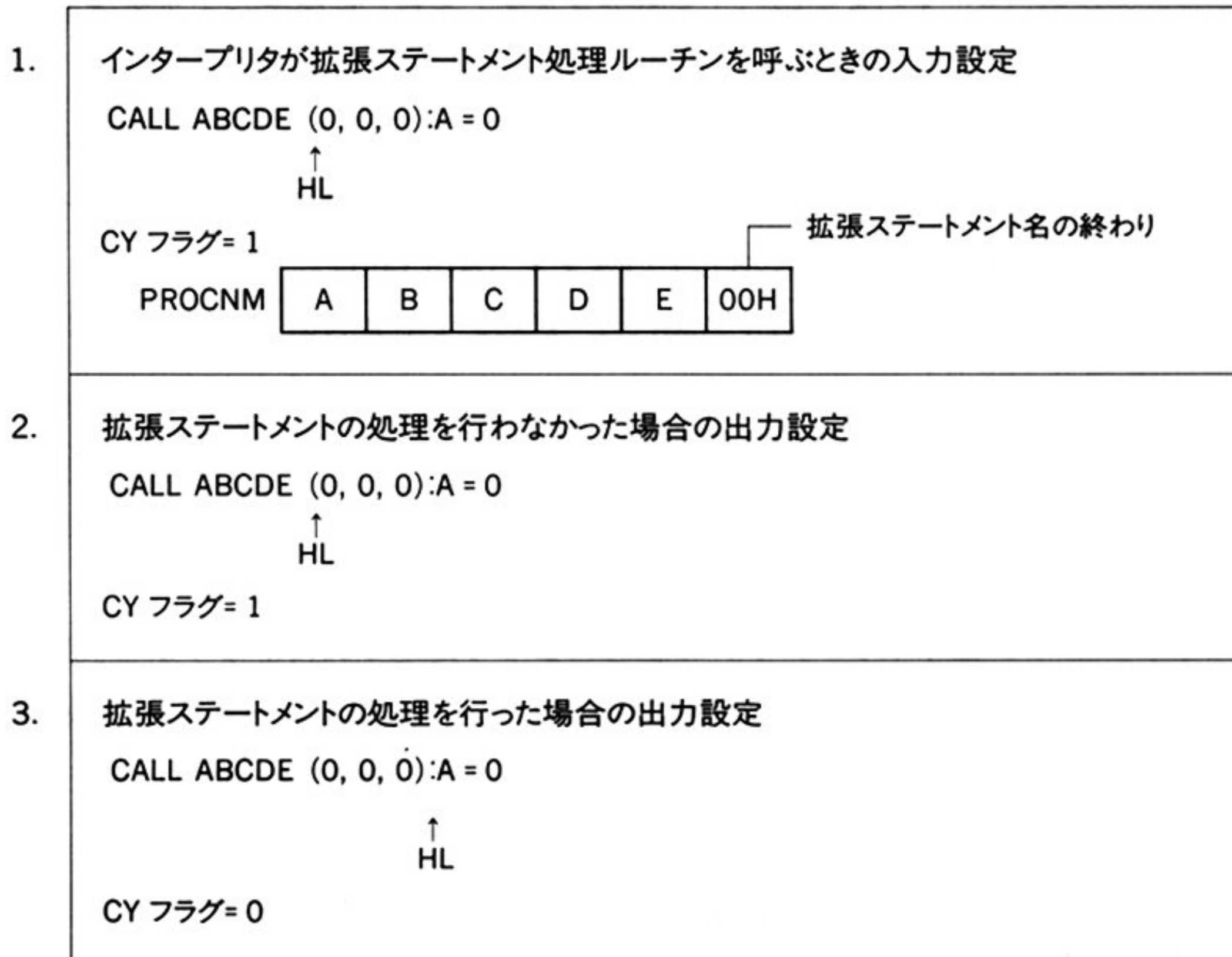



図5.13 拡張ステートメント処理ルーチンの入出力

ステートメント拡張ルーチンを作成する場合は、まずPROCNMに書かれた拡張ステートメント名を識別し、もしそれが自分の処理すべきものではなかった場合は、HLレジスタの変更はせずに、キャリーフラグを「1」にしてリターンします（図5.13-2)。拡張ステートメント名が自分の処理すべきものであった場合は、それに応じた処理をし、HLレジスタ（テキストポインタ）を自分が処理したステートメントの次（普通は00Hまたは3AHが入っているところ）に更新し、最後にキャリーフラグを「0」にしてリターンします（図5.13-3)。

BASICインタープリタはキャリーフラグの状態によって、CALL文が処理されたかどうかを判定し、処理されていない場合は次のカートリッジを呼び出します。すべてのカートリッジが処理をしなかった場合（最後までキャリーフラグが「1」だったとき）は「SYNTAX ERROR」を表示します。ステートメントの引数評価には「第2部 5.4 BASICの内部ルーチン」を利用すると便利です。

■ DEVICE

この2バイトには、カートリッジがデバイスの拡張（入出力装置の拡張）を行うのであれば、デバイス拡張ルーチンのアドレスを書き込み、行わなければ0000Hとしておきます。デバイスの拡張を行う場合、デバイス拡張ルーチンは4000H～7FFFHになければいけません。デバイスは1つのカートリッジについて、4つまで持つことができます。また、拡張デバイス名は15文字以下でなければいけません。

BASICインタープリタは定義されていないデバイス名を見つけると、【PROCNM(0FD89H, 16)】にそれを格納し、AレジスタにFFHを入れて、DEVICEの内容が0以外のカートリッジにスロット番号の小さい方から順に制御を渡していきます（図5.14-1）。

そこで、デバイス拡張ルーチンを作成する場合は、初めにPROCNMのファイルディスクリプタを識別し、もしそれが自分の処理すべきでないデバイスに対するものだった場合は、キャリーフラグを1にしてリターンして下さい(図5.14-2)。自分の処理すべきデバイスに対するものだった場合は、それに応じた処理を行い、装置ID(0～3)をAレジスタに設定し、最後にキャリーフラグを0にしてリターンして下さい（図5.14-3)。

BASICインタープリタはキャリーフラグの状態によって処理されたかどうかを判定し、もし処理されていない場合は次のカートリッジを呼び出します。すべてのカートリッジが処理をしなかった場合（最後までキャリーフラグが「1」だったとき）は「Bad file name」エラーを表示します。

実際の入出力操作をするときには、BASICインタープリタは【DEVICE(0FD99H)】に装置ID(0～3)を、Aレジスタにデバイスに対する要求（表5.1）を設定してデバイス拡張ルーチンを呼び出します。デバイス拡張ルーチンではそれを参照し、要求に応じた処理を行って下さい。

この拡張デバイス処理については、MSXの内部システムに関する深い知識が必要になるので、確固たる目的があるとき以外は、拡張ステートメントでデバイスをコントロールするのが現実的です。

1. インタープリタが拡張デバイス処理ルーチンを呼ぶときの入力設定

OPEN "ABC:" ・・・・・

Aレジスタ= FFH
CY フラグ= 1

ファイルディスクリプタの終わり

PROCNM	A	B	C	00H

2. 拡張デバイス処理を行わなかった場合の出力設定

CY フラグ= 1

3. 拡張デバイス処理を行った場合の出力設定

A レジスタ=装置 ID (0～3)
CY フラグ= 0

図 5.14　デバイス拡張ルーチンへの入出力

表 5.1　デバイスに対する要求

A レジスタ	要　求
0	OPEN
2	CLOSE
4	ランダムアクセス
6	シーケンシャル出力
8	シーケンシャル入力
10	LOC 関数
12	LOF 関数
14	EOF 関数
16	FPOS 関数
18	バックアップキャラクタ

■ TEXT

この2バイトには、カートリッジ内の BASIC プログラムをオートスタート（リセット時に実行）実行するのであれば、BASIC プログラムのテキストポインタを入れておき、行わなければ 0000H としておきます。プログラムのサイズは 8000H～BFFFH の 16K 以下でなければいけません。

BASIC インタープリタはイニシャライズ（INIT）を終え、システムを起動した後、ヘッダの TEXT の内容をスロット番号の小さい順に調べます。そして、そこが 0000H 以外であった場合、

そのアドレスをBASICテキストポインタとしてそこから実行を始めるようになっています（図5.15)。このときのBASICプログラムは、中間コード形式で格納されている必要があり、またBASICプログラムの初め（TEXTが指すアドレス）にはプログラムの開始を示す00Hが付いていなければいけません。

なお、GOTO文などの分岐先の行番号を、あらかじめ分岐先のテキストポインタの絶対アドレスにしておくと、プログラムの実行速度が上がります。

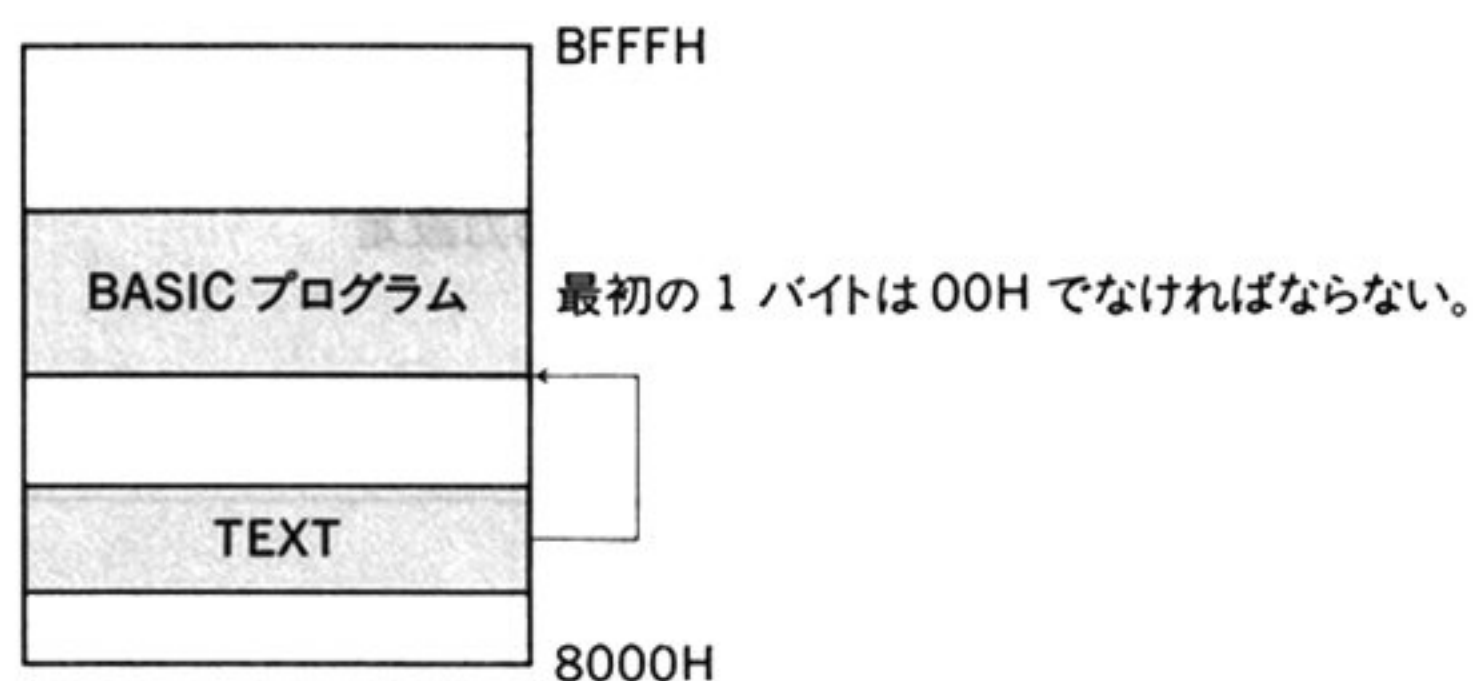



図5.15 BASICプログラムカートリッジの実行

BASICプログラムのROM化の方法

1. BASICのテキスト格納先頭アドレスを8021Hに変える。

```
POKE &HF676,&H21:POKE &HF677,&H80:POKE &H8020,0:NEW
```

注 意　必ず1行にまとめて実行する。

2. 目的のBASICプログラムをロードする。

```
LOAD "PROGRAM"
```

3. IDを作成する。

```
AD =&H8000
FOR I = 0  TO  31
          POKE  AD+I,0      ID エリアのクリア
NEXT I
POKE &H8000,ASC("A")
POKE &H8001,ASC("B")
POKE &H8008,&H20
POKE &H8009,&H80
```

4. 8000H〜BFFFHをそのままROMに焼き込む。

3.2 カートリッジ用ソフトの作成に関する諸注意

3.2.1 カートリッジで使用するワークエリアの確保

他のカートリッジに入っているプログラムといっしょに実行する必要がないプログラム（ゲームカートリッジのようなスタンドアロンのソフトウェア）では、BIOS の使用するワークエリア（F380H）よりアドレスの小さい部分は自由に使用することができます。

しかし、BASIC インタープリタの機能を利用して実行されるプログラムでは、同じ領域をワークエリアとして使用するわけにはいきません。その対策としては以下の 3 つの方法があります。

1. カートリッジ自体に RAM をのせてしまう（最も安全確実な方法）。
2. 必要なワークエリアが 1 バイトあるいは 2 バイトならば【SLTWRK(0FD09H～)】の自分に対応する 2 バイトをワークとして使用する。
3. 必要なワークエリアが 3 バイト以上ならば BASIC で使用する RAM から確保する。具体的には、まず【BOTTOM(0FC48H)】の内容を【SLTWRK(0FD09H～)】の対応するエリアに書き写し、BOTTOM の値を必要なワークエリア分増やし、そこをワークエリアとして確保する（図 5.16）。

0FFFFH
システムワーク
0F380H
ユーザーエリア
08000H
←(BOTTOM)

システムワーク
ユーザーエリア
←(BOTTOM)
ワークエリア
←(SLTWRK)

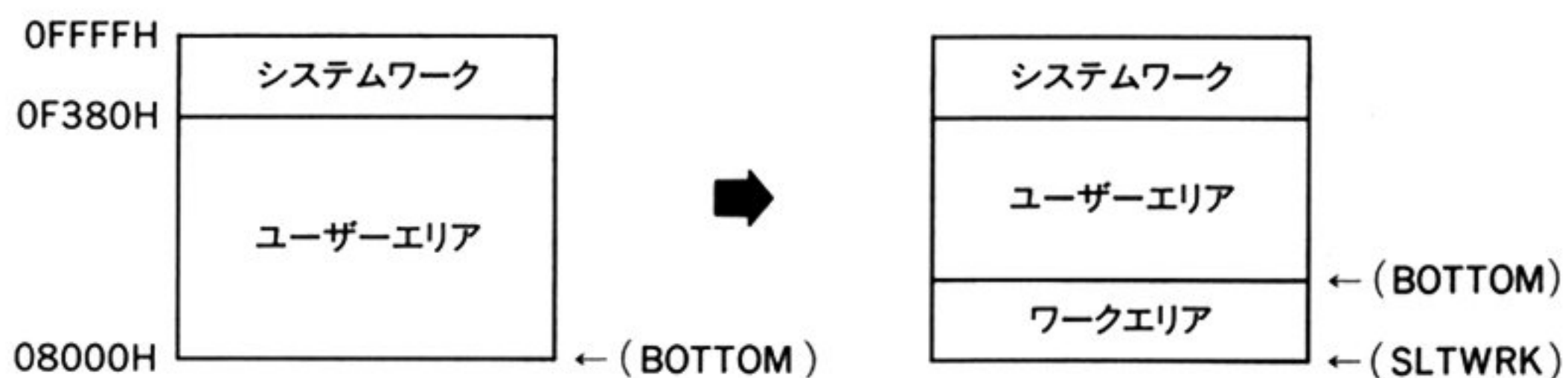



図 5.16　ワークエリアの確保

2.と 3.については添付のフロッピーディスクに入っている「GETWORK.MAC」を参考にして下さい。

3.2.2 フック

MSX-BASICが使用するワークエリアのFD9AH〜FFC9Hには「フック」というBASICの機能拡張のための領域があります。1つのフックは5バイトあり、そこには普通「RET」が書かれています。

MSX-BASICはある処理（ワークエリアのフックの説明で示すような処理）を行うと、そこから一度このフックをコールします。「RET」の場合はすぐ戻ってしまいますが、その5バイトをあらかじめイニシャライズ（INIT）ルーチンによりカートリッジ内のプログラムにインタースロットコールするように書き換えておけば、BASICの機能を拡張することができるわけです（図5.17参照）。

添付のフロッピーディスク内の「HOOK.MAC」は、割り込み処理用のH.KEYIというフックをカートリッジが利用する場合のプログラムです。

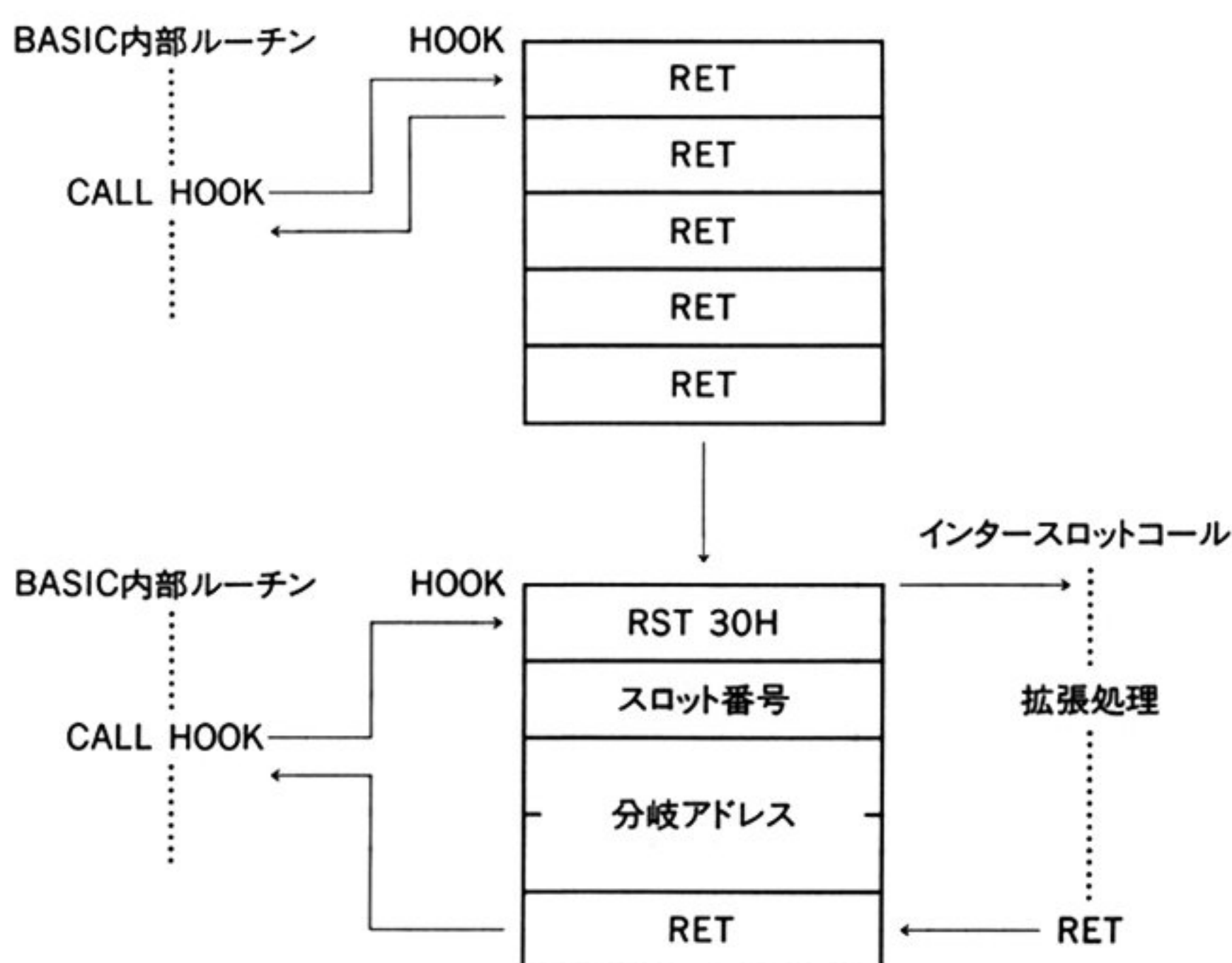



図5.17　フックの設定

3.2.3 スタックポインタの初期化

ディスク内蔵のMSXの場合、スロットの位置によってはカートリッジよりもディスクインターフェイスROMの方が先に初期化動作を行い、ワークエリア確保のためスタックポインタを

下位アドレス方向に下げることがあります。このとき、ディスクを使わないソフトウェアはカートリッジに制御が渡ってきてからもう一度スタックポインタを設定し直さないと、スタック領域がなくなり暴走などのトラブルを生じる可能性があります。プログラムの初めにはスタックポインタのイニシャライズを忘れずにおこなって下さい。

3.2.4 拡張スロットでの動作チェック

一般の市販ソフトウェアが拡張スロットに差し込まれた場合や RAM が拡張スロットにある場合に、アプリケーションプログラムが動作しないという問題があります。MSX2 の多くの機種は本体内部で拡張スロットを使用しているので、拡張スロットでの動作不良は致命的です。市販されるソフトウェアは、それが拡張スロットに入れられている場合と RAM が拡張スロットにある場合について、必ず動作チェックを行って下さい。

特に FFFFH 番地には拡張スロットレジスタが置かれますので、そこを RAM のつもりで参照してはいけません。例えば、プログラム中で「LD SP,0」を行って FFFFH 番地にスタックを設定すると、そのプログラムは拡張スロットを用いた機種では暴走します。

3.2.5 CALSLT 使用時の注意

CALSLT および CALLF でインタースロットコールを行うと、IX、IY、および裏レジスタの内容は破壊されます。また、このルーチンから返ってくるとき、MSX1 では割り込みが禁止されていますが、MSX2 では呼び出す前の状態に戻ります。

第6部
標準的な周辺装置のアクセス

MSXの基本的な考え方は、BIOSを通して周辺装置をアクセスすることにより、機種やバージョンの差異を吸収して互換性を高めようというものです。したがって、何をするにもまずBIOSの使用法を知っておかなければなりません。この第6部では、MSXの標準的な周辺装置をBIOSによりアクセスする方法と、そのために必要となる各周辺装置の構造を説明します。

1章 PSGと音声出力

MSXには次に示すような3系統の音声出力機能があります。ただし、3.はMSXに標準的に付属するものでないため、第7部で扱います。ここではそれ以外の1.と2.の機能について説明します。

1. PSGによる音声出力（3チャンネル、8オクターブ）
2. 1ビットI/Oポートによる音声出力
3. MSX-AUDIOやMSX-MUSICによる音声出力（FM音源）

1.1 PSGの機能

MSXの音楽演奏機能およびBEEP音の発生には、AY-3-8910相当のLSIが使われています。このLSIはPSG（Programmable Sound Generator）とも呼ばれ、その名が示すとおり、プログラムにより複雑な音楽やさまざまな効果音を発生することができます。その特徴をまとめると次のようになります。

1. 3チャンネルのトーン発生器を持ち、各チャンネルには独立に、4096種の音階（8オクターブに相当）と16段階の音量を指定することができる。
2. エンベロープパターンによって、ピアノやオルガンのような音色を出すことができる。ただし、エンベロープ発生器がひとつしか存在しないため、同時に使用できる音色は1種類だけである。
3. 内蔵するノイズ発生器によって、風の音や波の音のような効果音も容易に得られる。ただし、ノイズ発生器がひとつしか存在しないため、同時に使用できるノイズは1種類だけである。
4. 入力クロックfcを分周することで、トーンやエンベロープをはじめとする必要なすべての周波数を得ている（なおMSXでは、fc＝1.7897725MHzと定められている）。このため音程やリズムのふらつきがまったくない。

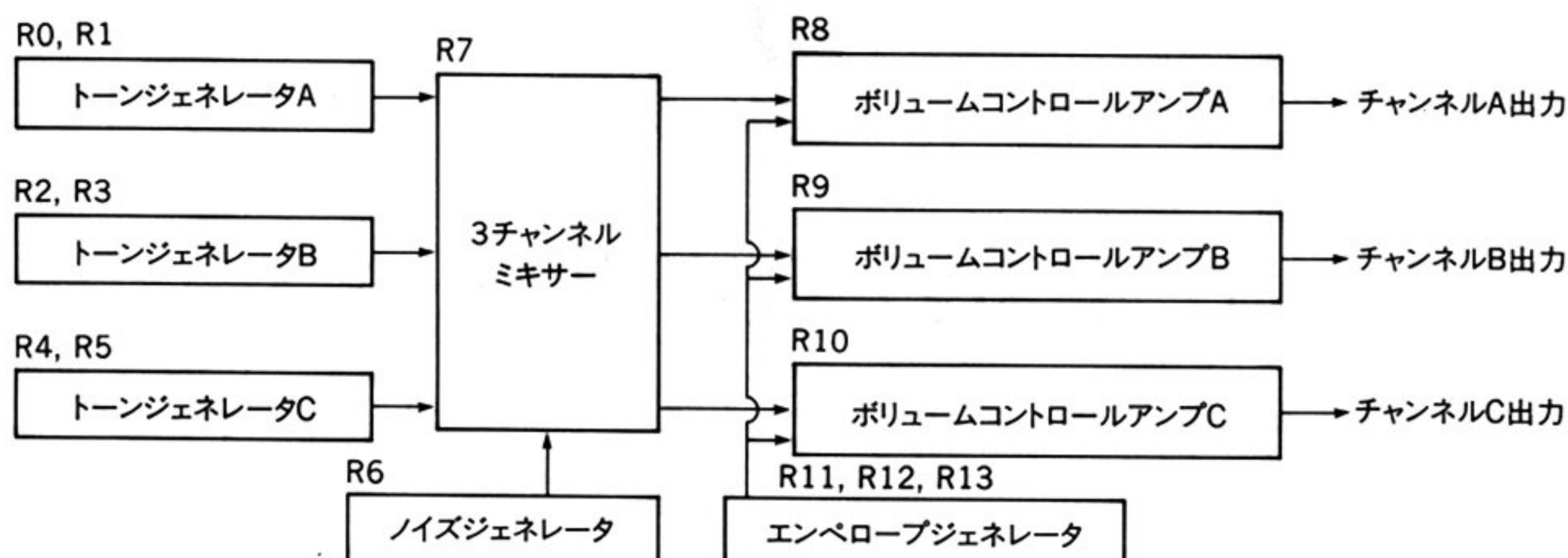



図 6.1　PSG のブロックダイアグラム

上のブロックダイアグラムでは省略しましたが、PSG は音声発生の機能とはまったく別に 2 つの入出力ポートを持っています。MSX ではこれを汎用入出力ポートとして、ジョイスティック、タッチパッド、パドル、マウスなどの入出力装置との接続にも利用しています。なお、これらの汎用入出力ポートについては、5 章で説明します。

1.1.1 PSG のレジスタ

PSG が音声出力の作業をすべて実行してくれるので、CPU の仕事は単に音が変化する時点で PSG にそれを教えることだけです。これは、図 6.2 に示したような PSG 内部の 16 個の 8 ビットレジスタに値を書き込むことによって行います。

以下に、これらのレジスタの役割とその使用法を説明します。

1. トーン周波数の設定(R0〜R5)

A、B、C 各チャンネルのトーン周波数は R0〜R5 によって設定します。まず PSG 内で入力クロック（fc = 1.7897725MHz）を 16 分周し、それを基準周波数とします。また各チャンネルは、この基準周波数をそれぞれに割り当てられた 12 ビットのデータで分周して最終的な音程を得ます。12 ビットの分周データ（TP）と発生する音の周波数（ft）には次のような関係があります。

```
ft = fc / (16 * TP)
   = 0.11186078125 / TP[MHz]
   = 111860.78125 / TP[Hz]
```

12ビットの分周データTPは、各チャンネルごとに上位4ビットの粗調整値CTと下位8ビットの微調整値FTによって、図6.3のように指定されます。また、音階を作るためのレジスタの設定を表6.1に示します。

レジスタ ＼ ビット		B7	B6	B5	B4	B3	B2	B1	B0
R0	チャンネルA 音程分周比	下位8ビット							
R1						上位4ビット			
R2	チャンネルB 音程分周比	下位8ビット							
R3						上位4ビット			
R4	チャンネルC 音程分周比	下位8ビット							
R5						上位4ビット			
R6	ノイズ分周比								
R7	$\overline{\text{Enable}}$	$\overline{\text{IN}}$/OUT		$\overline{\text{NOISE}}$			$\overline{\text{TONE}}$		
		IOB	IOA	C	B	A	C	B	A
R8	チャンネルA 音量				M				
R9	チャンネルB 音量				M				
R10	チャンネルC 音量				M				
R11	エンベロープ周期	下位8ビット							
R12		上位8ビット							
R13	エンベロープ波形								
R14	I/O ポートA								
R15	I/O ポートB								

図6.2 PSGのレジスタ構成

R0, R2, R4　8ビット
R1, R3, R5　4ビット
Coarse Tune (CT)　Fine Tune (FT)
TP
チャンネルA — R0, R1
チャンネルB — R2, R3
チャンネルC — R4, R5

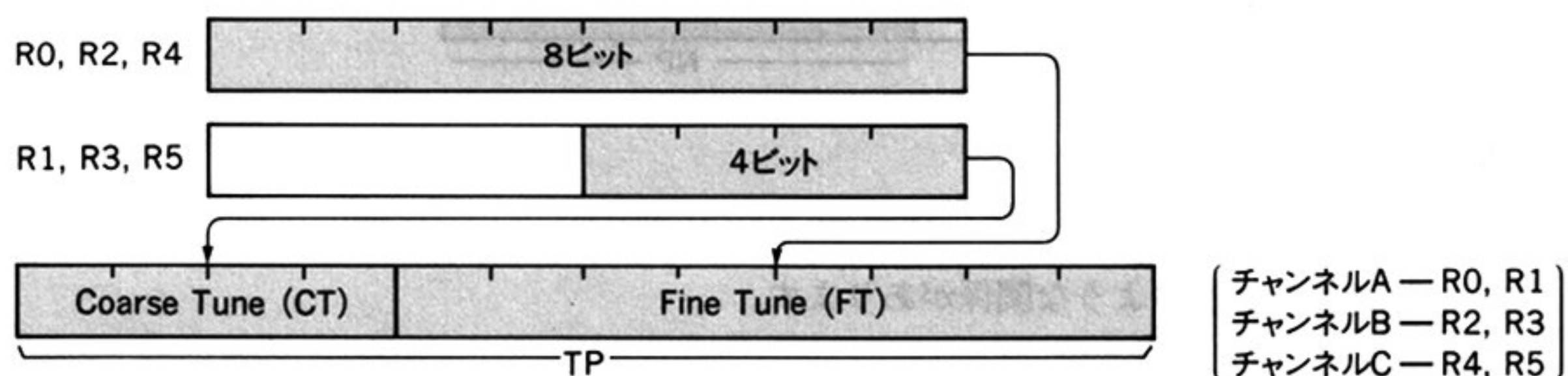



図6.3 音程の設定

表6.1 トーン周波数の設定(音階データ)

音程	オクターブ							
	1	2	3	4	5	6	7	8
C	D5D	6AF	357	1AC	D6	6B	35	1B
C#	C9C	64E	327	194	CA	65	32	19
D	BE7	5F4	2FA	17D	BE	5F	30	18
D#	B3C	59E	2CF	168	B4	5A	2D	16
E	A9B	54E	2A7	153	AA	55	2A	15
F	A02	501	281	140	A0	50	28	14
F#	973	4BA	25D	12E	97	4C	26	13
G	8EB	476	23B	11D	8F	47	24	12
G#	86B	436	21B	10D	87	43	22	11
A	7F2	3F9	1FD	FE	7F	40	20	10
A#	780	3C0	1E0	F0	78	3C	1E	F
B	714	38A	1C5	E3	71	39	1C	E

2. ノイズ周波数の設定(R6)

爆発音や波の音などの合成にかかせないのがノイズ音です。PSGはノイズジェネレータによって発生させたノイズをA～Cの各チャンネルに出力できます。ノイズジェネレータは1つしかないため、各チャンネル共通に同一のノイズが出力されます。ノイズは平均周波数を変えることによって様々な効果を出すことができ、R6レジスタがその設定を行います。このレジスタの下位5ビットのデータ(NP)で基準周波数(fc/16)を分周し、ノイズの平均周波数(fn)を決定します。

R6
NP

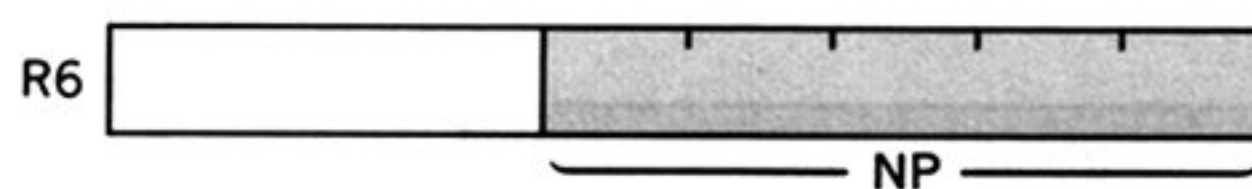



図6.4 ノイズ周波数の設定

NPとfnの間には次のような関係があります。

$$fn = fc/(16 * NP)$$
$$= 0.11186078125/NP[MHz]$$
$$= 111860.78125/NP[Hz]$$

NPは1〜31の値をとりますから、ノイズの平均周波数は3.6kHz〜111.9kHzの範囲で設定できます。

3. 音のミキシング(R7)

R7は各チャンネルごとに、トーンジェネレータとノイズジェネレータの出力を選択するためのレジスタで、ノイズとトーンの両者を混合することもできます。図6.5に示したように、R7の下位3ビット（B0〜B2）はトーン出力、次の3ビット（B3〜B5）はノイズ出力の制御を行います。どちらも対応するビットが0のときに出力ON、1のときにOFFとなります。

R7 | B7 | B6 | B5 | B4 | B3 | B2 | B1 | B0

Input Enable	
B	A

Noise Enable		
C	B	A

Tone Enable		
C	B	A

I/Oポート
入力——0
出力——1

ノイズ出力
ON ——0
OFF——1

トーン出力
ON ——0
OFF——1

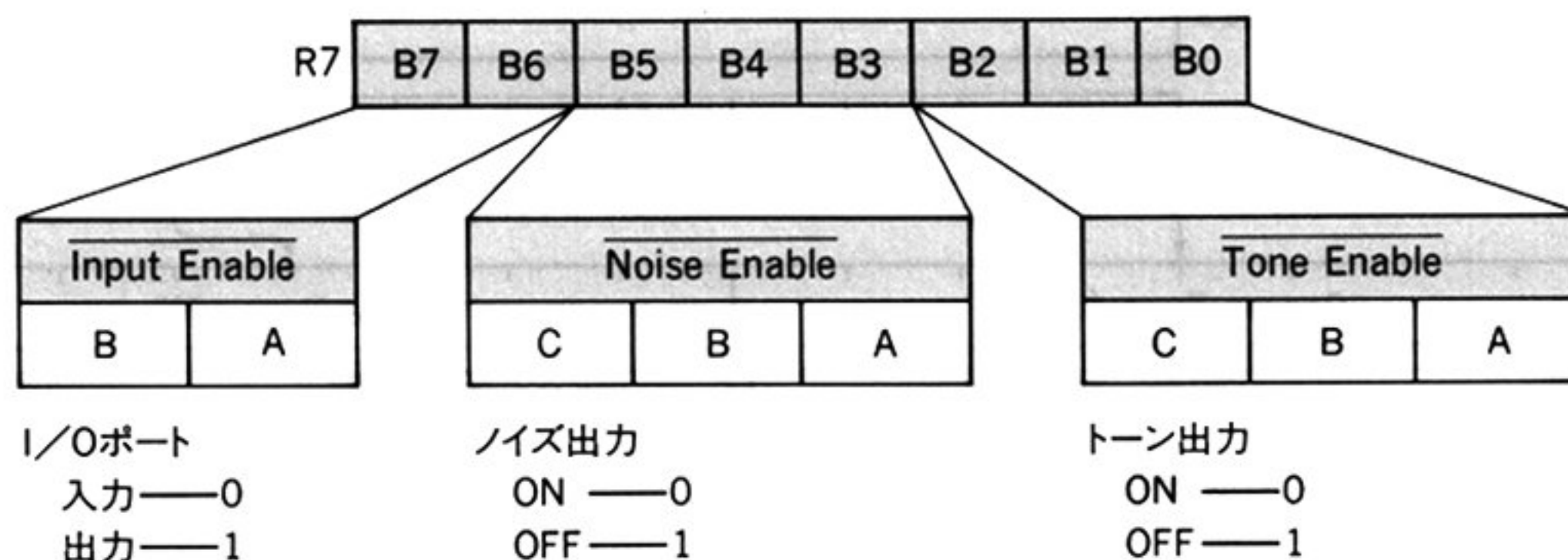



図6.5　各チャンネルの出力選択

R7の上位2ビットは音声出力とは関係ありません。これはPSGが持つ2本のI/Oポートのデータ方向を決定するもので、対応するビットが「0」のときに入力モード、「1」のときに出力モードが設定されます。MSXではポートAを入力、ポートBを出力として使用していますから、つねにビット6=「0」、ビット7=「1」と設定しておく必要があります。

4. 音量の設定(R8〜R10)

R8〜R10は各チャンネルの音量を指定するレジスタです。4ビットのデータ（0〜15）で音量を固定的に指定する方法とエンベロープを用いてビブラートや減衰音などの効果音を発生させる方法の2通りが選べます。その選択もこのレジスタで行います。

R8, R9, R10 | | | | B4 | B3 | B2 | B1 | B0
（B3〜B0：L）

エンベロープを
使用しない———0（Lの値で音量設定）
使用する————1（Lの値は無視）

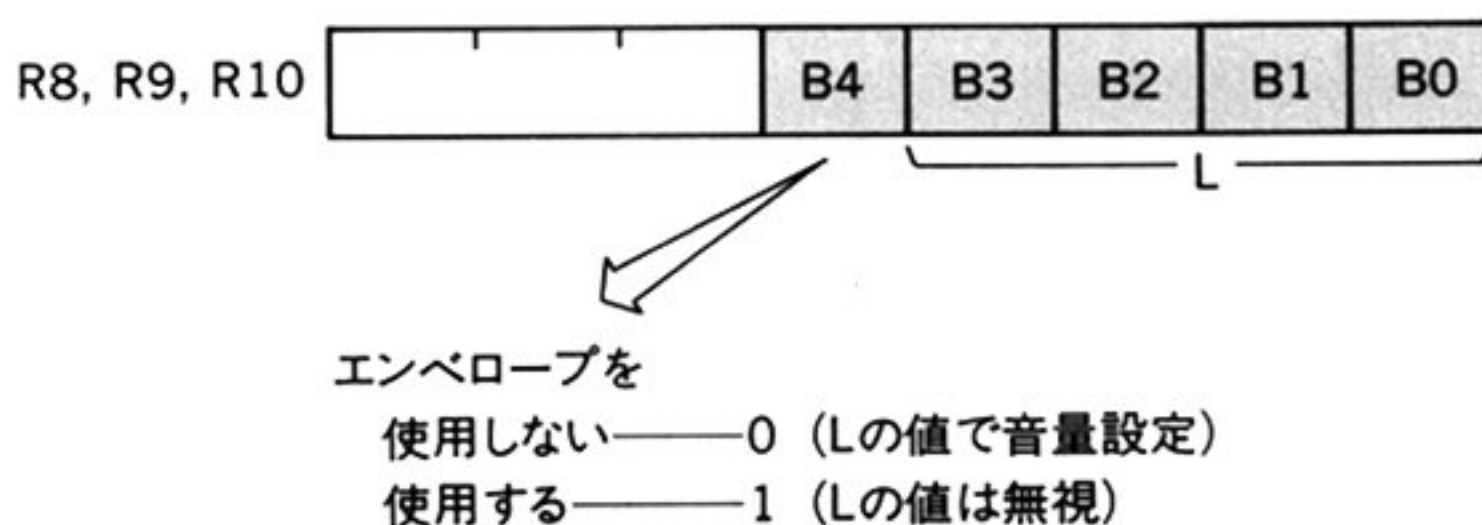



図6.6　音量の設定

なお、これらのレジスタのビット4が「0」の場合、エンベロープは使用されず、レジスタの下位4ビットの値L(0〜15)によって音量が指定されます。また、ビット4が「1」の場合はエンベロープ信号によって音量が変化し、Lの値は無視されます。

5. エンベロープ周期の設定(R11、R12)

R11、R12はエンベロープの周期を16ビットデータによって指定します。データの上位8ビットをR12で、下位8ビットをR11で設定します。

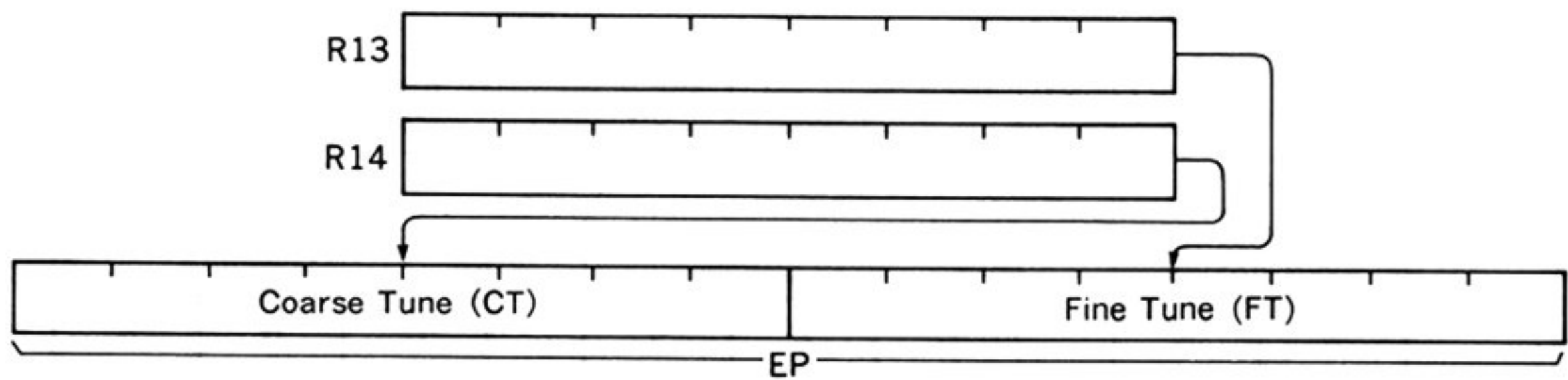



図6.7 エンベロープ周波数の設定

なお、エンベロープの周期Tと16ビットデータEPの間には、次のような関係があります。

T =(256＊EP) / fc
　=(256＊EP) / 1.787725[MHz]
　= 143.03493＊EP[μs]

6. エンベロープパターンの設定(R13)

R13は、下位4ビットのデータによって図6.8のようにエンベロープパターンを設定します。同図に示したTの間隔がR11とR12で指定されたエンベロープ周期に相当します。

R13

B3	B2	B1	B0
0	0	×	×
0	1	×	×
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1
1	1	0	0
1	1	0	1
1	1	1	0
1	1	1	1

T

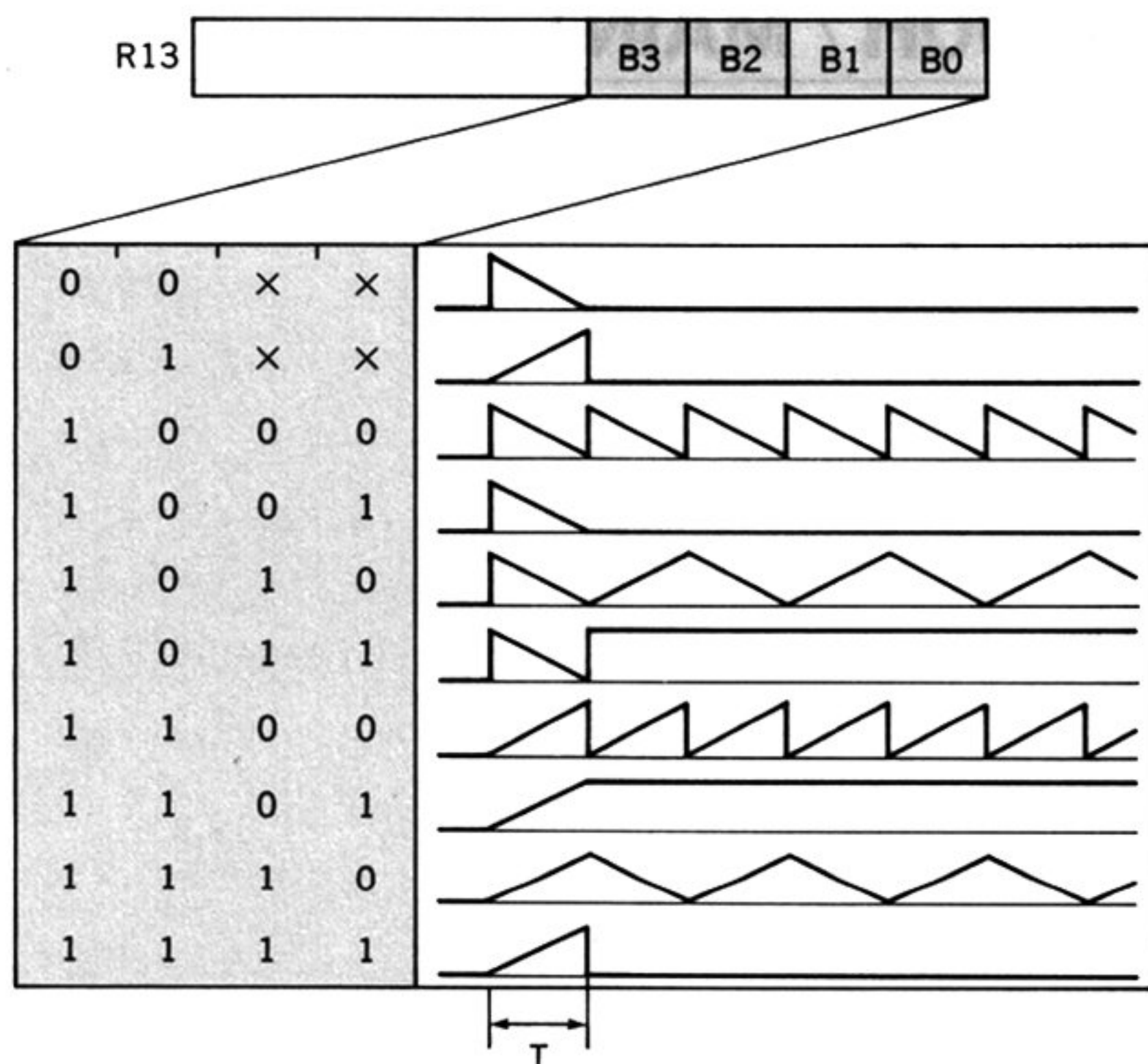



図 6.8　エンベロープの波形の設定

7. I/O ポート(R14、R15)

R14 と R15 は 8 ビットのデータをパラレルで入出力するポートです。MSX ではここを汎用入出力インターフェイスとして使用しています。詳しくは 5 章を参照してください。

1.2 PSG のアクセス

マシン語プログラム中から PSG をアクセスするために、以下のような BIOS ルーチンが用意されています。

GICINI(0090H / MAIN)

機　能

PSGを初期化します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

解　説

PSGのレジスタを初期化し、さらにBASICのPLAY文を実行するための作業領域の初期設定を行います。このとき、PSGの各レジスタは表6.2のような値に設定されます。

表6.2　PSGレジスタの初期値

レジスタ		ビット							
		7	6	5	4	3	2	1	0
R0	チャンネルA	0	1	0	1	0	1	0	1
R1	周波数	0	0	0	0	0	0	0	0
R2	チャンネルB	0	0	0	0	0	0	0	0
R3	周波数	0	0	0	0	0	0	0	0
R4	チャンネルC	0	0	0	0	0	0	0	0
R5	周波数	0	0	0	0	0	0	0	0
R6	ノイズ周波数	0	0	0	0	0	0	0	0
R7	チャンネル設定	1	0	1	1	1	0	0	0
R8	チャンネルA音量	0	0	0	0	0	0	0	0
R9	チャンネルB音量	0	0	0	0	0	0	0	0
R10	チャンネルC音量	0	0	0	0	0	0	0	0
R11	エンベロープ周期	0	0	0	0	1	0	1	1
R12		0	0	0	0	0	0	0	0
R13	エンベロープパターン	0	0	0	0	0	0	0	0
R14	I/OポートA								
R15	I/OポートB								

WRTPSG(0093H / MAIN)

機　能

PSG レジスタへデータを書き込みます。

コール手順

A　　PSG レジスタ番号
E　　書き込むデータ

戻り値

なし

変更レジスタ

なし

解　説

A レジスタで指定した番号の PSG レジスタに、E レジスタの内容を書き込みます。サンプルプログラム「TONE.MAC」を参照して下さい。

RDPSG(0096H / MAIN)

機　能

PSG レジスタのデータを読み出します。

コール手順

A　　PSG のレジスタ番号

戻り値

A　　指定したレジスタの内容

変更レジスタ

なし

解　説

A レジスタで指定した番号の PSG レジスタの内容を読み出し、その値を A レジスタに格納します。

STRTMS(0099H / MAIN)

機　能

音楽の演奏を開始します。

コール手順

(QUEUE)　中間言語に変換されたMML

戻り値

なし

変更レジスタ

すべて

解　説

バックグラウンドタスクとして、音楽が流れているかどうかを判定し、もし流れていなければキューに設定された音楽を演奏する。
実際のアクセスについては、「第2部 6.1 PLAY文BIOS」を参照して下さい。

1.3　1ビットサウンドポートによる音声発生機能

MSXには、PSGのほかにもうひとつ音源が存在しています。これは1ビットのI/Oポートの出力をソフトで繰り返しON/OFFして音を出すというものです。

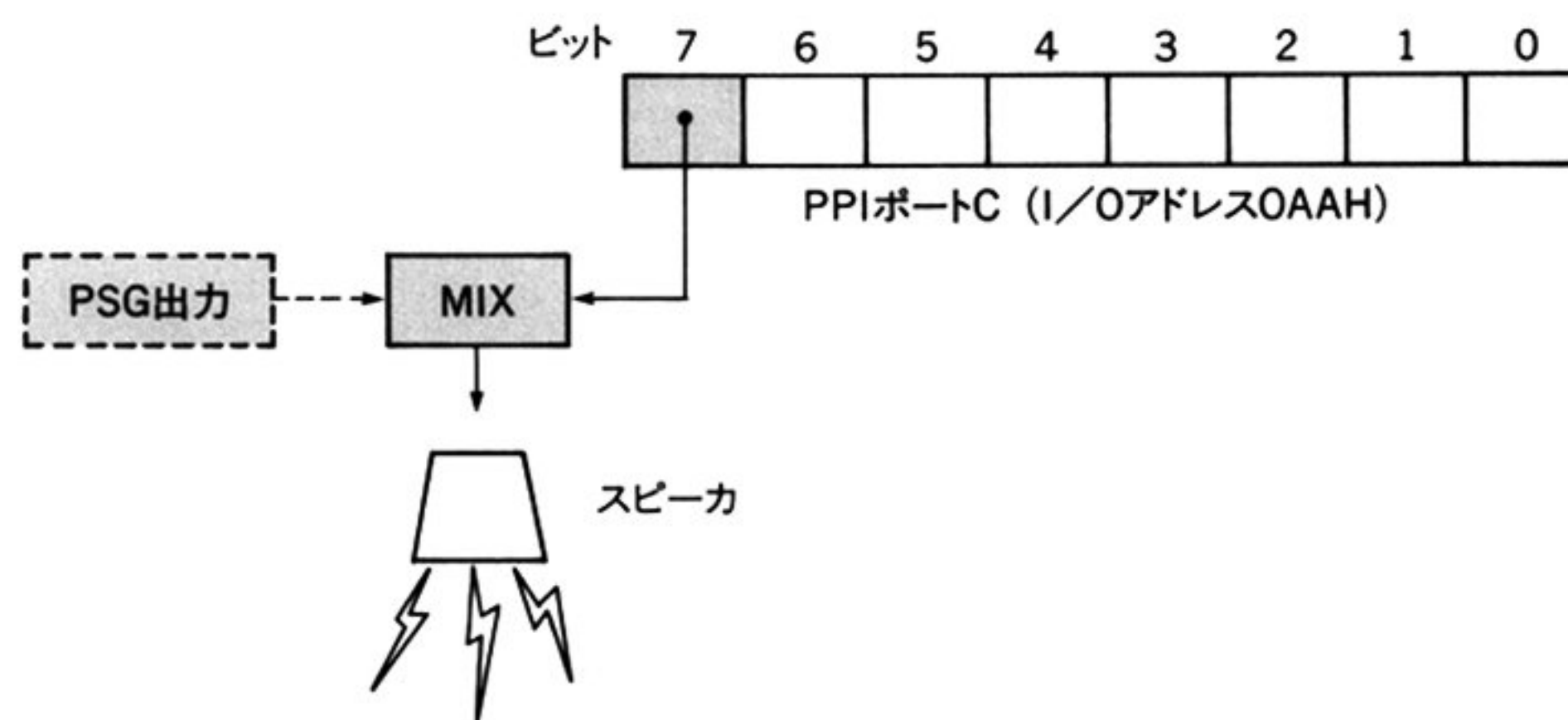



図6.9　1ビットサウンドポート

1.4 1ビットサウンドポートのアクセス

この1ビットサウンドポートをアクセスするために、以下のようなBIOSルーチンが用意されています。

CHGSND(0135H / MAIN)

機　能

1ビットサウンドポートをアクセスします。

コール手順

A　ON / OFF 指定
　0　OFF
　0以外　ON

戻り値

なし

変更レジスタ

AF

解　説

Aレジスタに0を入れてこのルーチンをコールするとサウンドポートのビットをOFFにし、0以外の値を入れてコールするとサウンドポートのビットをONにします。サンプルプログラム「CHGSND.MAC」を参照して下さい。

2章

カセットインターフェイス

カセットテープレコーダは、MSXの最も安価な外部記憶装置です。このカセットテープ上の情報をマシン語プログラム中で取り扱うためには、カセットインターフェイスの使用法を知らなくてはなりません。本章では、このために必要となる情報を説明します。

2.1 ボーレート

MSXのカセットインターフェイスは以下の2種類のボーレートを使用できます(表6.3)。なお、BASICのスタート時にはデフォルトで1200ボーが設定されています。

表6.3　MSXのボーレート

ボーレート	特　徴
1200bps	低スピード、高信頼性
2400pbs	高スピード、低信頼性

ボーレートの指定は、SCREEN命令の第4パラメータまたはCSAVE命令の第2パラメータで行います。一度指定すると以後そのボーレートが引き続いて使用されます。

```
SCREEN ,,,＜ボーレート＞
CSAVE "ファイル名",＜ボーレート＞
```

(＜ボーレート＞は両者とも、「1」で1200ボー、「2」で2400ボー)

2.2 1ビットの構成

入出力の基本となる1ビットのデータは図6.10のように記録されます。パルスの幅はCPUのT-STATEをカウントして決められるため、カセットインターフェイス作動中はすべての割り込みが禁止されています。

なお、カセットから入力されるビットデータは、汎用入出力インターフェイスのポートB(PSGのレジスタ# 15)の第7ビットを通して読み取ることが可能です。サンプルプログラム「CFILES.MAC」の中で、この機能を使用していますので、参照して下さい。

ボーレート	ビット	波　形
1200ボー	0	(1200Hz×1波)
	1	(2400Hz×2波)
2400ボー	0	(2400Hz×1波)
	1	(4800Hz×2波)

2983 T-states (833μsec)
1491 T-states (417μsec)
746 T-states (208μsec)
373 T-states (104μsec)

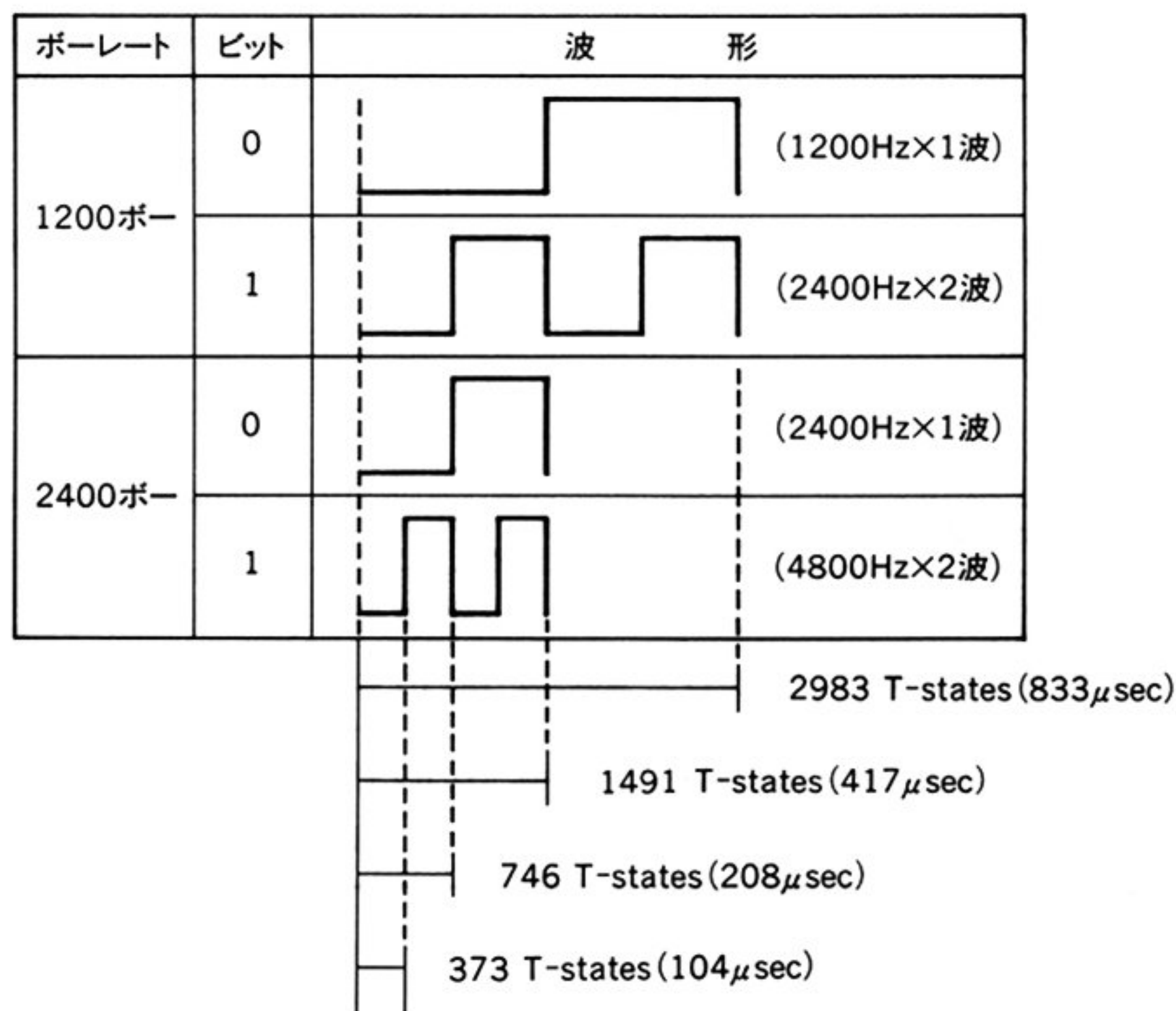



図6.10　1ビットの構成

2.3 1バイトの構成

1バイトのデータは図6.11のようなビット列で記録されます。スタートビットとして「0」のビットがひとつ、次にデータ本体が最下位ビットから最上位ビットの順で8ビット、最後にストップビットとして「1」のビットが2つ続き、合計11ビットを使用します。

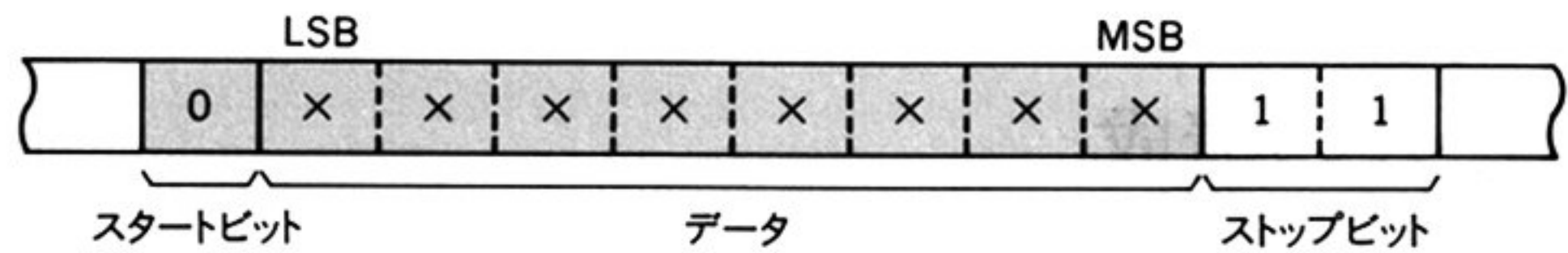



図6.11 1バイトの構成

2.4 ヘッダの構成

ヘッダとはセーブまたはロード時に、カセットテープが動き出してから回転が安定するまでの時間待ちや、ファイル間に区切りをつけるために、ある特定の周波数の信号を一定時間テープに記録させた部分のことです。ロングヘッダとショートヘッダの2種類があり、ロングヘッダはモータの回転が安定するまでの時間待ちに使用されます。また、テープリード時のボーレートは、ロングヘッダを読んで決定されます。ショートヘッダはファイルボディ間の区切りに使用されます。表6.4に両者の構成を示しました。

表6.4 ヘッダの構成

ボーレート	ヘッダの種類	ヘッダの構成
1200ボー	ロングヘッダ	2400Hz×16000波（約6.7秒）
	ショートヘッダ	2400Hz× 4000波（約1.7秒）
2400ボー	ロングヘッダ	4800Hz×32000波（約6.7秒）
	ショートヘッダ	4800Hz× 8000波（約1.7秒）

2.5 ファイルのフォーマット

MSXのBASICは次の3タイプのカセットフォーマットのファイルをサポートしています。

1. BASICテキストファイル

CSAVE命令でバイナリセーブしたBASICプログラムは、このフォーマットにしたがって記録されます。なお、ファイルは前半のファイルヘッダと後半のファイルボディに分けられます。

6.7秒
10バイト
6バイト
ロングヘッダ
0D3H×10個
ファイル名
ファイルヘッダ
ファイルボディ
ショートヘッダ
BASICプログラム
00H×7個
1.7秒
任意長
7バイト

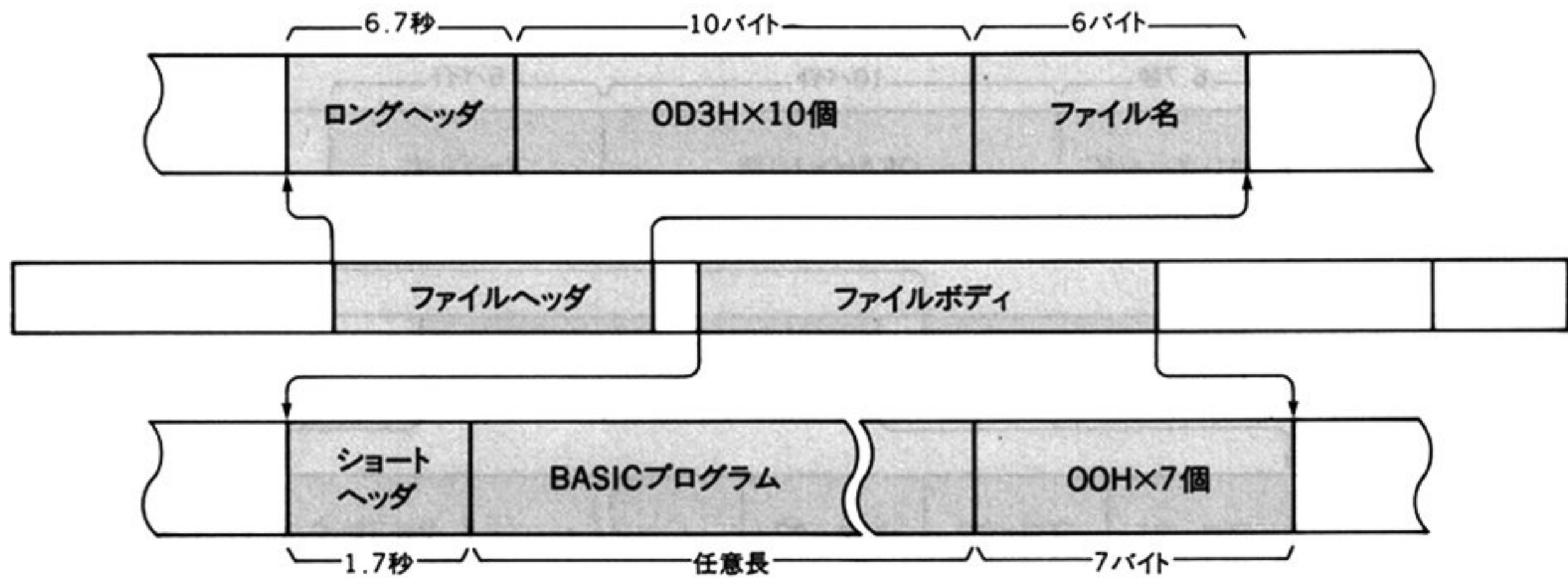



図6.12　バイナリファイルのフォーマット

ファイルヘッダは、ロングヘッダの後に0D3Hという値が10バイト続き、その後に6バイトのファイル名が置かれます。ファイルボディはショートヘッダの後にプログラム本体が続き、7バイトの00Hでファイルエンドを示します。

2. ASCIIテキストファイル

SAVE命令でアスキーセーブされたBASICプログラム、およびOPEN命令によって作られたデータファイルは、このフォーマットにしたがって記録されます。

ファイルヘッダは、ロングヘッダの後に0EAHという値が10バイト続き、その後に6バイトのファイル名が置かれます。

データ本体は256バイトごとのブロックに分けられ、それぞれのブロックの前にはショートヘッダが置かれます。ファイルエンドはCtrl-Z(1AH)で示されます。このため、1AHという値を含むデータファイルを作ることはできません。

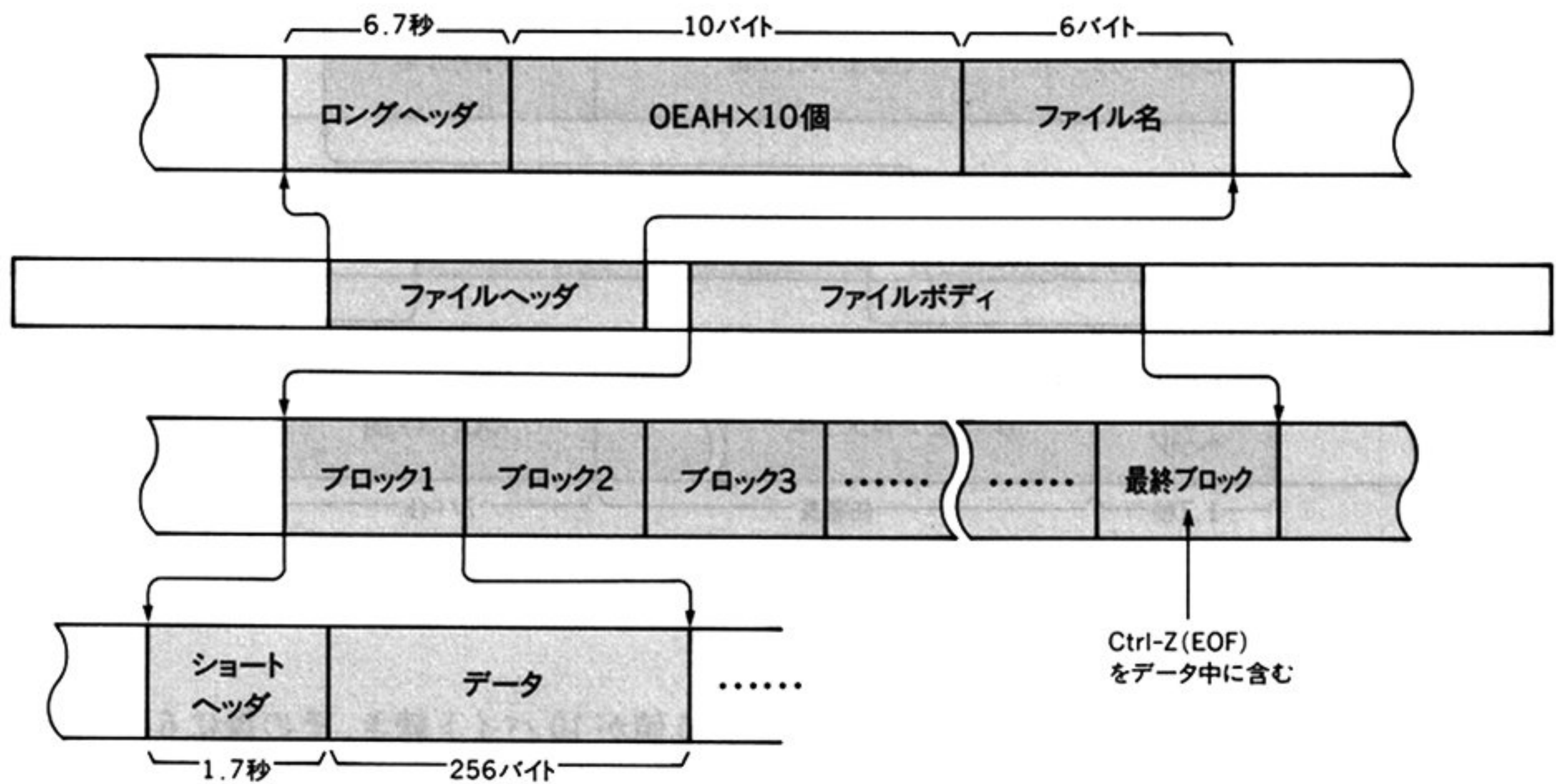



図 6.13 アスキーファイルのフォーマット

3. マシン語ファイル

BSAVE 命令でセーブされたマシン語ファイルは、このフォーマットにしたがって記録されます。ファイルヘッダは、ロングヘッダの後に 0D0H という値が 10 バイト続き、その後ろに 6 バイトのファイル名が置かれます。

ファイルボディは、ショートヘッダの後に、先頭アドレス、最終アドレス、実行開始アドレスの順に 3 つのアドレスが記録され、その次にマシン語本体が続きます。データの量は先頭アドレスと最終アドレスから計算できますから、特別なファイルエンドマークはありません。実行開始アドレスとは、BLOAD 命令の R オプション使用時に、プログラムが実行されるアドレスです。

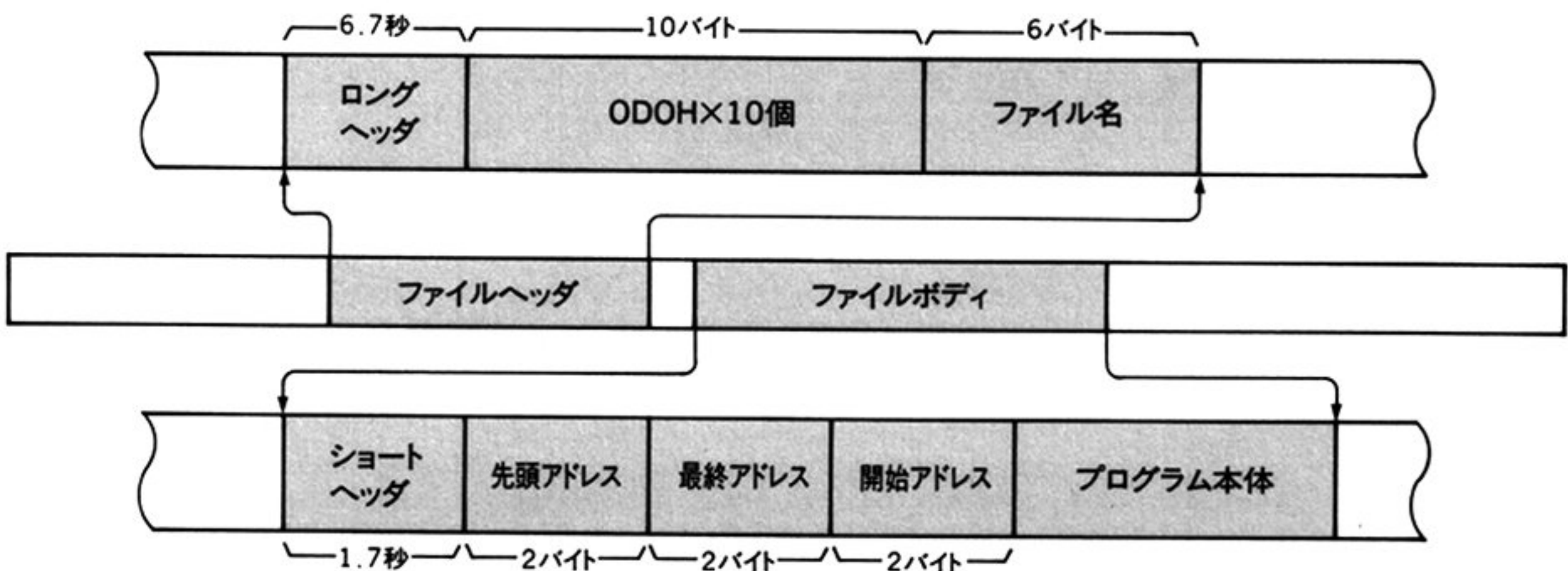



図 6.14 マシン語ファイルのフォーマット

2.6 カセットファイルのアクセス

カセットファイルをアクセスするために、以下の BIOS ルーチンが用意されています。

TAPION(00E1H / MAIN)

機　能

読み込み用にファイルをオープンします。

コール手順

なし

戻り値

異常終了時は CY フラグ ON

変更レジスタ

すべて

解　説

テープレコーダのモータを起動し、ロングヘッダまたはショートヘッダを読み込みます。同時にそのファイルが記録されたボーレートを判別し、ワークエリアをそれに合わせて設定します。割り込みは禁止されます。

TAPIN(00E4H / MAIN)

機　能

データを 1 バイト読み込みます。

コール手順

なし

戻り値

A　　　読み込んだデータ

異常終了時はCYフラグON

変更レジスタ

すべて

解　説

テープからデータを1バイト読み込み、Aレジスタに格納します。

TAPIOF(00E7H / MAIN)

機　能

読み込み用のファイルをクローズします。

コール手順

なし

戻り値

なし

変更レジスタ

なし

解　説

テープからの読み込み動作を終了します。同時に割り込みは再開されます。

TAPOON(00EAH / MAIN)

機　能

書き出し用にファイルをオープンします。

コール手順

A　　ヘッダの種類

0　　ショートヘッダ

0以外　ロングヘッダ

戻り値

異常終了時はCYフラグON

変更レジスタ

すべて

解　説

テープレコーダのモータを起動し、Aレジスタで指定された種類のヘッダをテープに書き出します。なお、割り込みは禁止されます。

TAPOUT(OOEDH / MAIN)

機　能

データを1バイト書き出します。

コール手順

A　　書き込みデータ

戻り値

異常終了時はCYフラグON

変更レジスタ

すべて

解　説

Aレジスタの内容をテープに書き込みます。

TAPOOF(00F0H / MAIN)

機　能

書き込み用のファイルをクローズします。

コール手順

なし

戻り値

なし

変更レジスタ

なし

解　説

テープへの書き込み動作を終了します。同時に割り込みは再開されます。

STMOTR(00F3H / MAIN)

機　能

モーターの動作を指定します。

コール手順

A　動作の指定

0	停止
1	起動
255	現在と反対の状態

戻り値

なし

変更レジスタ

AF

解　説

A レジスタで指定した値にしたがって、モーターの動作状態を設定します。

これらの BIOS を使用してカセットファイルの READ/WRITE ルーチンを作成する際、なるべく無駄な動作をせずに、ただひたすら READ または WRITE のみに専念するようにしてください。例えば、テープからデータを読み込みながらそのデータを CRT に表示させたりすると、READ エラーを生じることがあります。

サンプルプログラム「CFILES.MAC」を参照して下さい。

3章 キーボードインターフェイス

MSX2のキーボードはMSX1と同じ構造をしていますが、カナの入力時にローマ字カナ変換方式が使えるようになり、機能的にはたいへん便利になりました。この章ではMSX2のキーボードインターフェイスについて説明します。

なお、キーの配列などについては日本仕様のキーボードをもとに説明しますが、海外仕様のMSXではデータが少し異なる部分もあります。

3.1 キー配列

MSXのキーボードの配列は、英数字はJIS標準配列を採用しており、カナはJIS標準配列と五十音順配列をジャンパ線により切り換えています。ただし、このジャンパ線による設定はシステム起動時にどちらの配列を選ぶか決めているだけであり、ワークエリアKANAMDの値で任意に変更可能です。

【KANAMD(FCADH,1)】　キー配列（0＝五十音配列、0以外＝JIS配列）

3.2 キースキャン

MSXは図6.15のようなキーマトリクスを持っています。このキーマトリクスを調べることによって、キーの押されている状況をリアルタイムに得ることが可能であり、ゲームなどの入力に使用できます。

キーマトリクスのスキャンは、次のBIOSルーチンを用いて行います。

SNSMAT(0141H / MAIN)

機　能

キーマトリクスの指定行を読み出します。

コール手順

A　　読み出すキーマトリクスの行（0〜10）

戻り値

A　　指定したキーマトリクスの行の状態（押されているキーのビットが0になる）

変更レジスタ

AF、C

解　説

図6.15のキーマトリクスの1行を指定し、その状態をAレジスタに返します。このとき、押されているキーに対応するビットが「0」、押されていないキーに対応するビットは「1」になります。

サンプルプログラム「KEYSCAN.MAC」を参照して下さい。

行	7	6	5	4	3	2	1	0
0	' 7土 ニ	& 6金 ナ	% 5木ォ オ	$ 4水ェ エ	# 3火ゥ ウ	〃 2月ィ イ	! 1日ァ ア	0万 ノ
1	+ ;♣ モ	{ [○「 ロ	' @ レ	| ¥円 ル	~ ^ リ	= −ー ラ	) 9千 ネ	(8百 ヌ
2	B ┘ ト	A サ	_◆・ ン	? /♠。 ヲ	> .大、 ワ	< ,小ョ ヨ	}]●」 °	* :♥ー 〃
3	J ミ	I │ フ	H時 マ	G ┤ ソ	F ┼ セ	E ク	D ス	C └ ッ ツ
4	R ┤ ケ	Q カ	P π ホ	O ヘ	N ャ ヤ	M分 ユ	L中 メ	K ム
5	Z タ	Y年 ハ	X× チ	W キ	V ┴ テ	U ヒ	T ┐ コ	S秒 シ
6	F8 F3	F7 F2	F6 F1	かな	CAPS LOCK	GRAPH	CTRL	SHIFT
7	RETURN	SELECT	BS	STOP	TAB	ESC	F10 F5	F9 F4
8	→	↓	↑	←	DEL	INS	CLS HOME	SPACE

[TEN KEY]

行	7	6	5	4	3	2	1	0
9	4	3	2	1	0	option	option	option
10	.	,	−	9	8	7	6	5

図6.15 キーマトリクス

3.3 文字の入力

MSXは、タイマ割り込みにより1/60秒ごとにキーマトリクスをスキャンしており、キーが押されていれば、その文字コードを図6.16のようなリング状のキーボードバッファに格納します。なお、MSXのキー入力は、一般にこのキーボードバッファを読むことによって行っています。

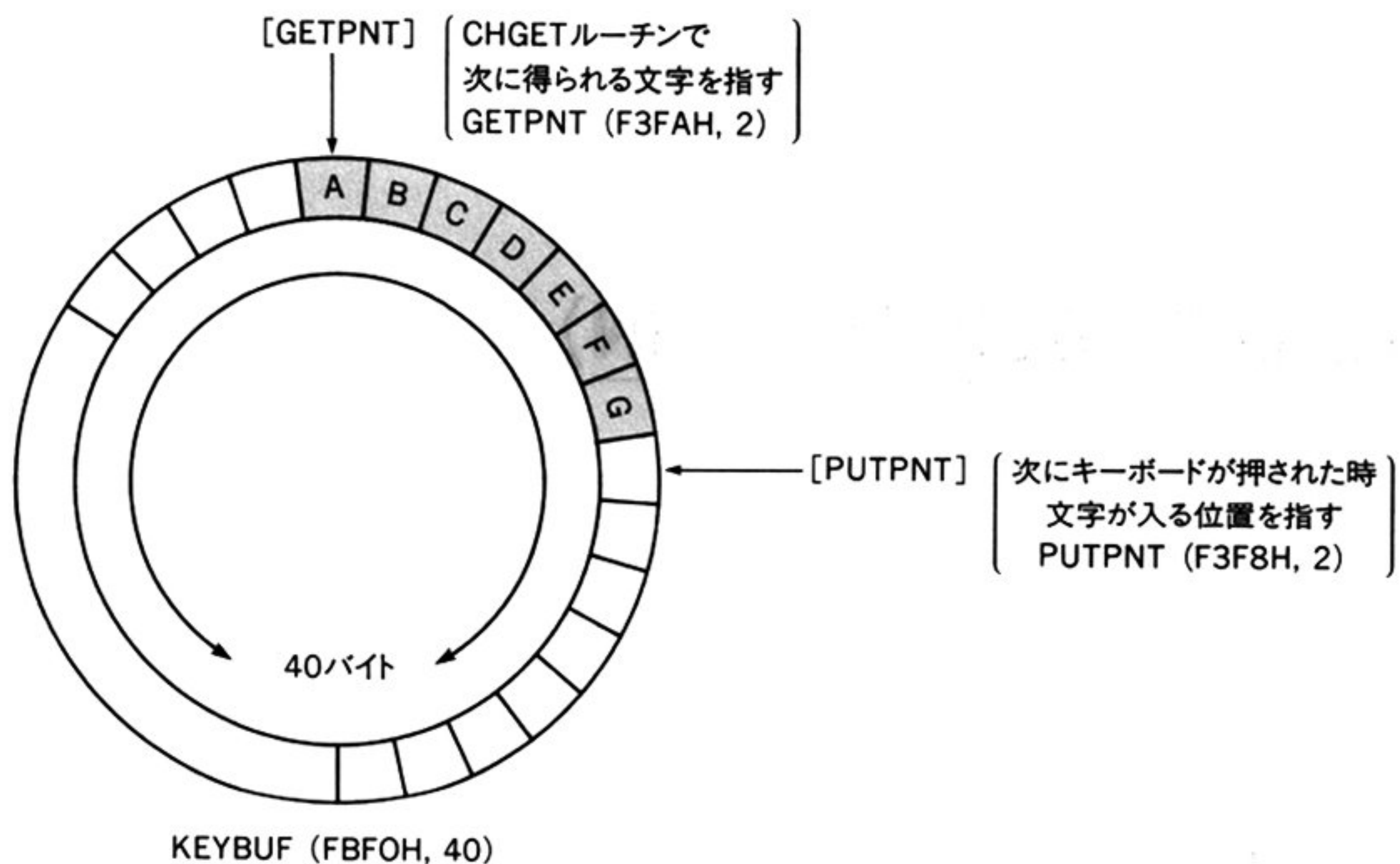



図 6.16　リング状のキーボードバッファ

このキーボードバッファを使用したキー入力およびそれに関連する機能を持った BIOS を以下に示します。タイマ割り込みを禁止すると、当然これらは使用できなくなります。

CHSNS(009CH / MAIN)

機　能

キーボードバッファの残りをチェックします。

コール手順

なし

戻り値

バッファが空なら、Z フラグ ON

変更レジスタ

AF

解　説

キーボードバッファに文字が残っているかどうかを調べます。キーボードバッファが空であればZフラグを立てます。

CHGET(009FH / MAIN)

機　能

キーボードバッファから文字を1文字入力します。

コール手順

なし

戻り値

A　　文字コード

変更レジスタ

AF

解　説

キーボードバッファから1文字読み出してAレジスタに格納します。バッファに文字が入っていない場合には、カーソルを表示してキー入力があるまで待ちます。キー入力待ちの間、CAPロック、カナロック、ローマ字カナ変換ロックも有効です。関係のあるワークエリアは以下に示すとおりです。このうち、SCNCNTとREPCNTはCHGETルーチン実行後に初期化されてしまうので、オートリピートの時間間隔を変えるためにはCHGETコールのたびにこのワークを設定する必要があります。

【CLIKSW(F3DBH,1)】　キークリック音（0＝OFF、0以外＝ON）
【SCNCNT(F3F6H,1)】　キースキャンの時間間隔（通常は1）
【REPCNT(F3F7H,1)】　オートリピート開始までの時間（通常は50）
【MODE(FAFCH,1)】　ローマ字カナ変換のモード（図6.17参照）
【CSTYLE(FCAAH,1)】　カーソルの形（0＝■、0以外＝_）
【CAPST(FCABH,1)】　CAPSロック（0＝OFF、0以外＝ON）
【KANAST(FCACH,1)】　カナロック（0＝OFF、0以外＝ON）

【MODE(FAFCH,1)】

7	6	5	4	3	2	1	0
	•	•	•	•			

VRAM 容量（ビット 2〜1）

ビット 7: 0 ひらがなモード / 1 カタカナモード

ビット 0: 0 通常カナ入力 / 1 ローマ字カナ変換

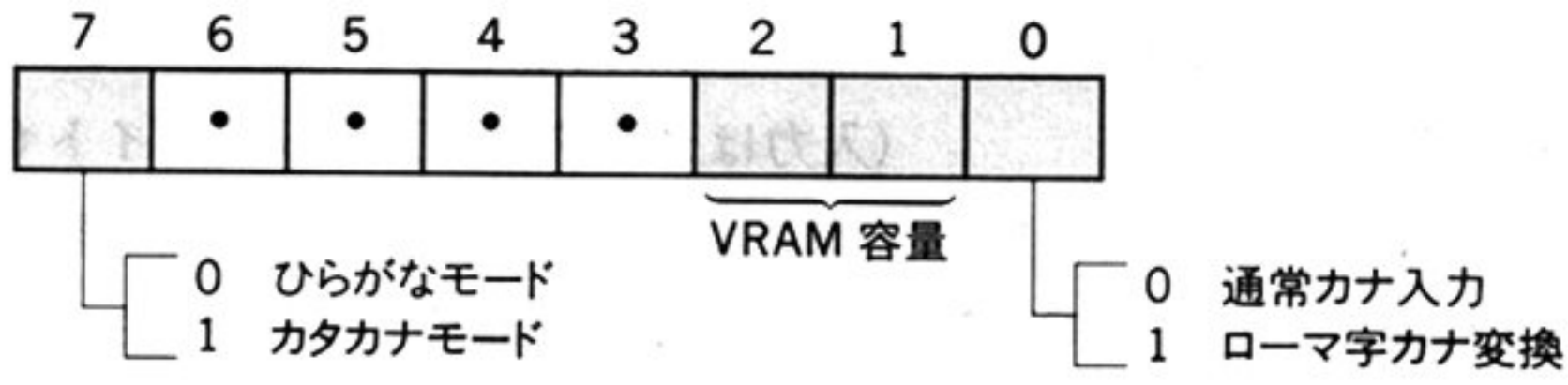



図 6.17　ローマ字カナ変換モードの設定

KILBUF(0156H / MAIN)

機　能

キーボードバッファを空にします。

コール手順

なし

戻り値

なし

変更レジスタ

HL

解　説

キーボードバッファを空にします。

サンプルプログラム「GETKEY.MAC」を参照して下さい。

CNVCHR(00ABH / MAIN)

機　能

グラフィック文字を処理します。

コール手順

A　　文字コード

戻り値

A　　グラフィック文字は変換される（通常文字ならば変換されない）
CY フラグ=OFF　　（入力はグラフィックヘッダバイト 01H だった）
CY フラグ=ON、Z フラグ=ON　　（入力はグラフィック文字なので変換された）
CY フラグ=ON、Z フラグ=OFF　　（入力は通常の文字なので変換されなかった）

変更レジスタ

AF

解　説

CHGET の後にこの CNVCHR を実行すると、グラフィック文字は表 6.5 のような 1 バイトコードに変換し、グラフィック文字以外は無変換でそのまま返します。グラフィック文字はグラフィックヘッダバイト (01H) を伴う 2 バイトの変則的なコードで表されるため文字処理のときにめんどうな手続きが必要ですが、このルーチンによって多少は手間が省けます。

PINLIN(00AEH / MAIN)

機　能

文字列を 1 行入力します。

コール手順

なし

戻り値

HL　　F55DH
F55EH　入力した文字列（行末は 00H で示される）
CY フラグ　　ON　　STOP で終了した
　　　　　　OFF　　RETURN で終了した

表 6.5 グラフィック文字のコード変換表

変換前	変換後	変換前	変換後
0141H (月)	41H	0151H (┴)	51H
0142H (火)	42H	0152H (┬)	52H
0143H (水)	43H	0153H (┤)	53H
0144H (木)	44H	0154H (├)	54H
0145H (金)	45H	0155H (┼)	55H
0146H (土)	46H	0156H (│)	56H
0147H (日)	47H	0157H (─)	57H
0148H (年)	48H	0158H (┌)	58H
0149H (円)	49H	0159H (┐)	59H
014AH (時)	4AH	015AH (└)	5AH
014BH (分)	4BH	015BH (┘)	5BH
014CH (秒)	4CH	015CH (□)	5CH
014DH (百)	4DH	015DH (大)	5DH
014EH (千)	4EH	015EH (中)	5EH
014FH (万)	4FH	015FH (小)	5FH
0150H (π)	50H		

変更レジスタ

すべて

解　説

入力した文字列をラインバッファ【BUF(F55DH,258)】に格納します。文字列の入力時にはスクリーンエディットのすべての機能が有効です。RETURN キーまたは STOP キーを押すと入力動作を終了します。なお、ワークエリアは以下に示すとおりです。

【BUF(F55DH,258)】　文字列が格納されるラインバッファ
【LINTTB(FBB2H,24)】 物理的 1 行が、上の行の継続であれば 00H

INLINE(00B1H / MAIN)

機　能

文字列を 1 行入力します（プロンプト使用可）。

コール手順

なし

戻り値

PINLINと同じ

変更レジスタ

すべて

解　説

PINLINルーチンと同様に、入力した文字列をラインバッファ【BUF(F55DH,258)】に格納します。ただし、こちらはルーチン実行開始時のカーソル位置より前の部分は入力されません。

両者の違いは、サンプルプログラム「INPIN.MAC」を参照して下さい。

3.4 ファンクションキー

MSXには10個のファンクションキーがあり、ユーザーが自由に定義して使うことができます。ファンクションキーの定義を行うために、ワークエリアが各キーについて16バイトずつ割り当てられています。そのアドレスを以下に示します。

【FNKSTR(F87FH,16)】	F1キーの定義用領域
【+10H(F88FH,16)】	F2キーの定義用領域
【+20H(F89FH,16)】	F3キーの定義用領域
【+30H(F8AFH,16)】	F4キーの定義用領域
【+40H(F8BFH,16)】	F5キーの定義用領域
【+50H(F8CFH,16)】	F6キーの定義用領域
【+60H(F8DFH,16)】	F7キーの定義用領域
【+70H(F8EFH,16)】	F8キーの定義用領域
【+80H(F8FFH,16)】	F9キーの定義用領域
【+90H(F90FH,16)】	F10キーの定義用領域

1つのファンクションキーを押すと、それぞれの領域に定義された文字列が【KEYBUF(FBF0H,40)】に格納されます。文字列の最後は00Hで示され、最大15文字まで定義可能です(16文字を超えた分は、複数のファンクションキー定義領域にまたがって定義される)。ファンクショ

ンキーを初期設定の状態に戻すには、次の BIOS ルーチンを利用するとよいでしょう。

INIFNK(003E / MAIN)

機　能

ファンクションキーを初期化します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

解　説

ファンクションキーの定義を BASIC スタート時の設定に戻します。

3.5 割り込み中の STOP キー

3.3 で説明した 1 文字入力ルーチン CHGET は、どのキーが押されているのかの判断をタイマ割り込みの処理ルーチン内で行っています。したがって、たとえば、カセットデータの入出力時などタイマ割り込みが禁止されている状態では、どんなキーが押されていても読み取ることができません。しかし、次に述べる BIOS ルーチンを使用すると、割り込みが禁止されている場合でも、CTRL + STOP キーが押されていることを判定できます。

BREAKX(00B7H / MAIN)

機　能

CTRL + STOP キーが押されているかどうかを判定します。

コール手順

なし

戻り値

CTRL + STOP が押されていれば、CY フラグ ON

変更レジスタ

AF

解　説

キースキャンを行い、CTRL キーと STOP キーが同時に押されているか否かを判定します。もし両方のキーが押されていれば、CY フラグを「1」にセットして戻ります。CTRL + STOP が押されていなければ、CY フラグを「0」にリセットして戻ります。このルーチンは、割り込みが禁止された状態でコールして下さい。

4章 プリンタインターフェイス

本章では、MSX のプリンタインターフェイスをマシン語からアクセスする方法について説明します。ここで述べる情報は、特にビットイメージ印字によってグラフィックを表示するような目的でプリンタを使用する場合に必要となってきます。

4.1 プリンタインターフェイスの概要

MSX は仕様により 8 ビットパラレルの出力ポートで、BUSY と STROBE 信号によるハンドシェーク方式でプリンタを動かします。コネクタなども仕様に定められています（アンフェノール 14 ピン、本体側メス）。信号線表を図 6.18 に示します。

プリンタ・インターフェイス
ピン接続

⑦⑥⑤④③②①
⑭⑬⑫⑪⑩⑨⑧

①……………$\overline{\text{STROBE}}$
②〜⑨………データ（b0〜b7）
⑪……………BUSY
⑭……………GND

×	×	×	×	×	×	×	×

I/O ポート（91H）
データ

・	・	・	・	・	・	・	×

I/O ポート（90H:WRITE 時）
$\overline{\text{STROBE}}$（「0」でデータ送出）

・	・	・	・	・	・	×	・

I/O ポート（90H:READ 時）
0 プリンタ READY
1 プリンタ BUSY

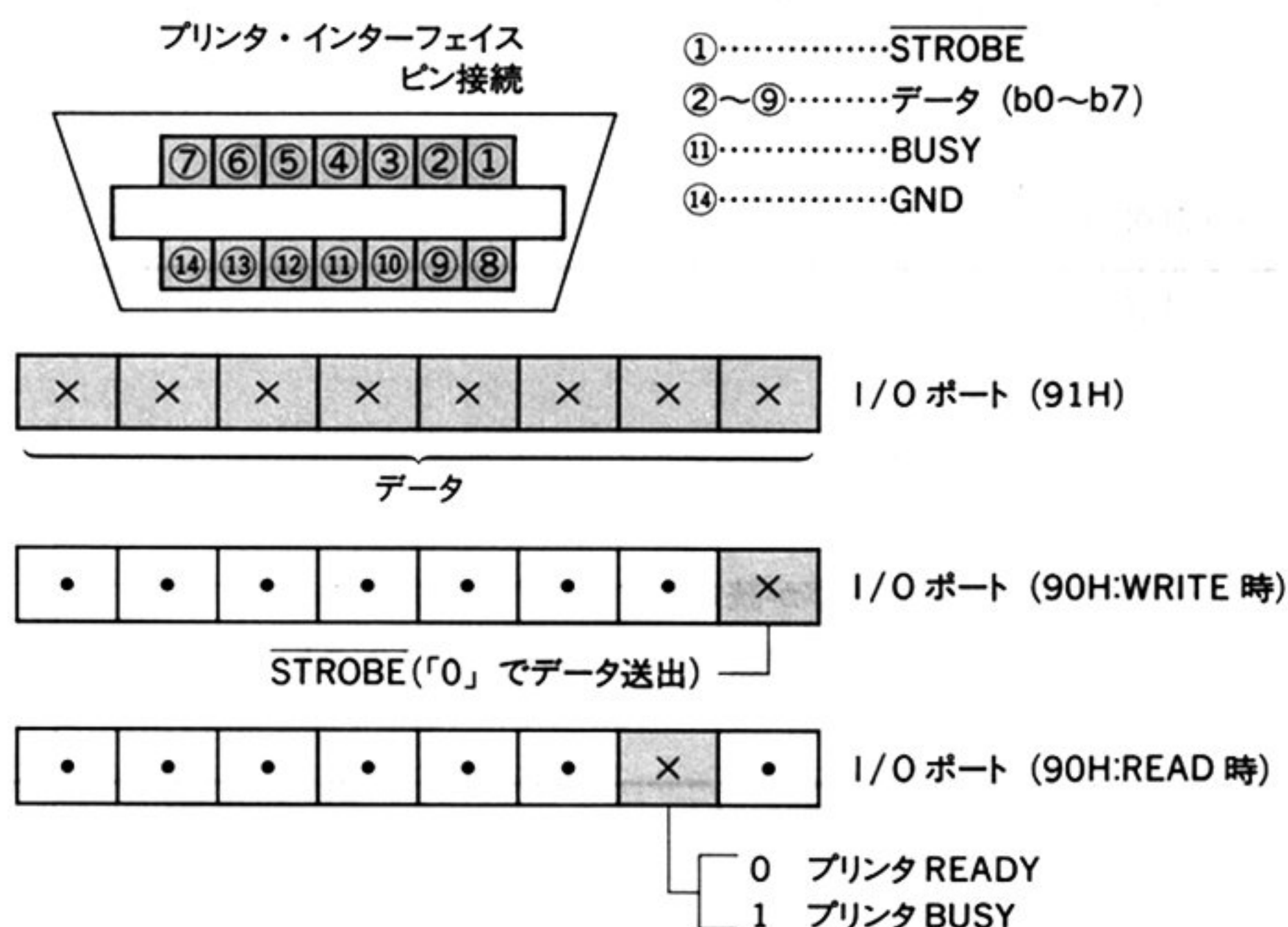



図 6.18 プリンタインターフェイス

4.2 MSX仕様のプリンタへの出力

MSXからプリンタへデータを送る場合、送られる側のプリンタがMSX仕様であるか否かによって、その動作は異なります。そこで、まずMSX仕様のプリンタの使い方について説明します。MSX仕様以外のプリンタについては次の節で述べることにします。

MSX仕様のプリンタは、MSXが画面に表示できるすべてのキャラクタが印字でき、キャラクタコードn=01H〜1FHに相当するグラフィックキャラクタ（月、火、水・・・）も、グラフィックキャラクタヘッダ（01H）の後に40H+nというコードを出力することによって印字可能です。さらに、MSX仕様のプリンタは、少なくとも、表6.6に示す制御コードが使用できます（漢字印字など、これ以外の機能を備えたプリンタの制御に関しては、それぞれのプリンタの説明書を参照）。

MSX仕様のプリンタを改行させるためには、0DHと0AHを続けて出力してください（MSX仕様でないプリンタには、0DHのみで改行する機種もある）。ビットイメージ印字を行うには、ESC+"Snnnn"（nnnnは10進4桁の数字）というエスケープシーケンスの後にnnnnバイトのデータを出力します。ただし、MSXはタブ機能を持たないプリンタのために、タブコード（09H）を適当な数のスペースコード（20H）に変換してプリンタに送る機能があり、通常は常にこの変換を行ってしまいます。09Hという値を含むビットイメージを正しく表示するためには、次のワークエリアを書き換えてください。

【RAWPRT(F418H,1)】　内容が00Hならばタブをスペースに置き換え、00H以外ならばそのまま出力する。

表6.6　プリンタ制御コード

コード	機能
0AH	ラインフィード
0CH	フォームフィード
0DH	キャリッジリターン
ESC+A	通常の改行間隔（文字が読みやすいように行間をあける）
ESC+B	グラフィック用の改行間隔（行間の隙間がなくなる）
ESC+Snnnn	ビットイメージ印字

4.3 MSX 仕様以外のプリンタへの出力

MSX 仕様ではないプリンタを使用する場合に問題となるのは、「ひらがな」をどう扱うかという点です。普通は、カタカナは印字できても、ひらがなは印字できないというプリンタが一般的です。MSX には、そのようなプリンタのために、ひらがなをカタカナに変換して出力する機能があります。BASIC からは、SCREEN 命令の第5パラメータで指定しますが、これは次のワークエリアを書き換えることによっても可能です。

【NTMSXP(F417H,1)】 内容が 00H ならば、ひらがなをカタカナに変換、00H 以外ならば、ひらがなをそのまま出力。

4.4 プリンタのアクセス

プリンタへの出力を行うために、以下に示す BIOS ルーチンが用意されています。プリンタの制御については、「第7部 6章 24ドット漢字プリンタ」を参照して下さい。

LPOUT(00A5H / MAIN)

機　能

プリンタへデータを出力します。

コール手順

A　　データ

戻り値

異常終了時は CY フラグ ON

変更レジスタ

F

解　説

Aレジスタで指定したデータをプリンタに出力します。

LPTSTT(00A8H / MAIN)

機　能

プリンタの状態を獲得します。

コール手順

なし

戻り値

A　　プリンタの状態

変更レジスタ

AF

解　説

現在のプリンタの状態をチェックします。このルーチンを呼び出して、Aレジスタが255でかつZフラグが0ならばプリンタは使用可能で、Aレジスタが0でかつZフラグが1ならば使用できません。

OUTDLP(014DH / MAIN)

機　能

プリンタへデータを出力します。

コール手順

A　　データ

戻り値

異常終了時はCYフラグON

変更レジスタ

F

解　説

Aレジスタで指定したデータをプリンタに出力します。LPTOUTルーチンとの相違点は次のとおりです。

1. TABコードは相当する個数のスペースをプリントします。
2. MSX仕様でないプリンタでひらがなを出力する場合、カタカナに変換します。
3. 異常終了した場合、「Device I/O error」となって返ってきます。

5章 汎用入出力インターフェイス

1章でも説明したように、MSXの使用しているPSGは、音声出力の機能とは別にポートAとポートBという2つの8ビット入出力ポートを持っています。MSXはこの2つのポートを汎用入出力インターフェイス（いわゆるジョイスティックポート）に接続して、ジョイスティックやマウスなどの装置とのデータ入出力に利用しています(図6.19)。この汎用入出力インターフェイスに接続される各種の装置は、それぞれ専用のBIOSルーチンが用意されており、手軽にアクセスすることができます。

ここでは、各入出力装置の機能とBIOSルーチンによるアクセス法について説明します。

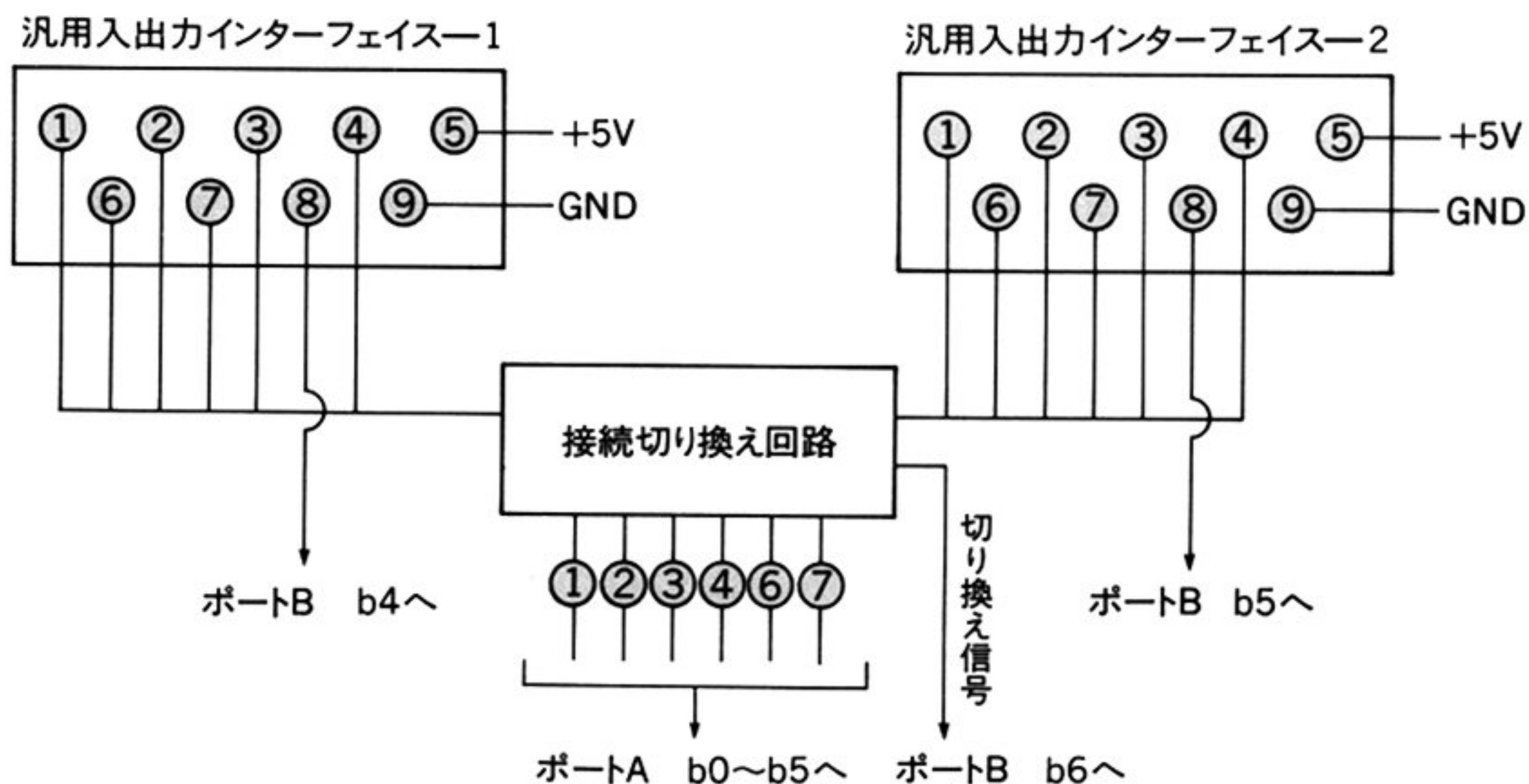



図6.19　汎用入出力インターフェイス

5.1 ポートの機能

PSGの2つの入出力ポートは、図6.20のように利用されています。

ポートA(PSG＃14)

入力

7	6	5	4	3	2	1	0

- bit0: ①番端子
- bit1: ②番端子
- bit2: ③番端子
- bit3: ④番端子
- bit4: ⑥番端子
- bit5: ⑦番端子

（bit0〜5）汎用入出力インターフェイスに接続される

- bit6: 0 50音配列キーボード / 1 JIS配列キーボード
- bit7: カセットからのデータ入力

ポートB(PSG＃15)

出力

7	6	5	4	3	2	1	0

- bit0: 汎用入出力インターフェイス1の6番端子に接続
- bit1: 汎用入出力インターフェイス1の7番端子に接続
- bit2: 汎用入出力インターフェイス2の6番端子に接続
- bit3: 汎用入出力インターフェイス2の7番端子に接続
- bit4: 汎用入出力インターフェイス1の8番端子に接続
- bit5: 汎用入出力インターフェイス2の8番端子に接続
- bit6: 0 ポートAのbit0〜5が汎用入出力インターフェイス1と接続 / 1 ポートAのbit0〜5が汎用入出力インターフェイス2と接続
- bit7: 0 カナモード表示ランプ点灯 / 1 カナモード表示ランプ消灯

図6.20 PSGのポートAとポートBの機能

5.2 ジョイスティックの使用法

ジョイスティックの回路を図6.21に示します。この回路からわかるように、8番端子に「0」を出力し、1〜4、6〜7の端子を読み出せば、スティックとトリガボタンの情報は得られますが、将来に渡る互換性を考慮すると、ジョイスティックのアクセスはやはりBIOSを利用して行う方が安全です。

ジョイスティックをアクセスするためには、以下に示すBIOSルーチンが用意されています。なお、これらのルーチンはBASICのSTICK関数やSTRIG関数とほぼ等しい機能を持っているもので、ジョイスティック以外にカーソルキーやスペースキーの状態をリアルタイムで読み取ることも可能です。

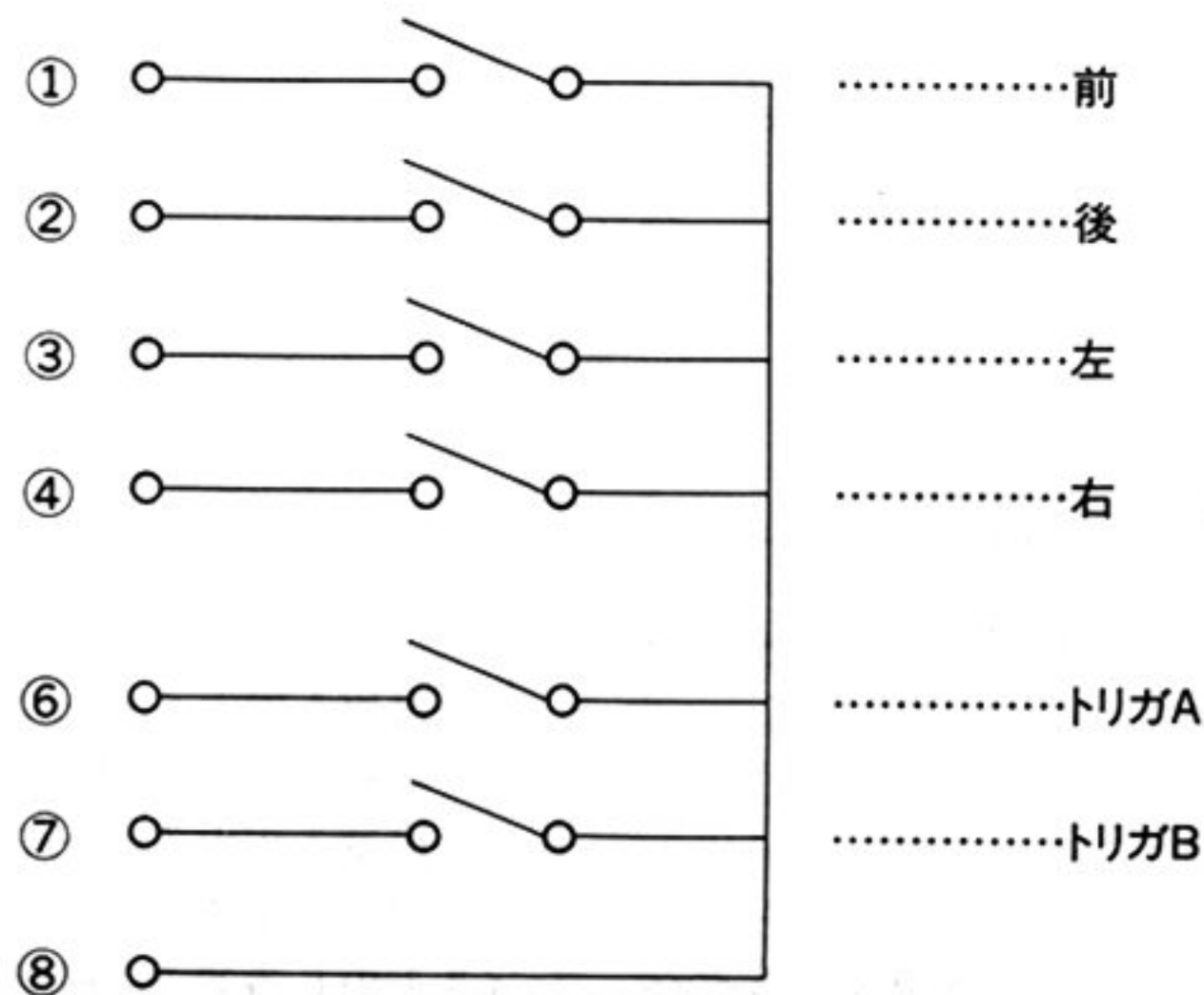



図6.21 ジョイスティック回路図

GTSTCK(00D5H / MAIN)

機 能

ジョイスティックの状態を読み出します。

コール手順

A　　ジョイスティック番号
　　0　　カーソルキー
　　1〜2　トリガボタン

戻り値

A　　ジョイスティックまたはカーソルキーの押された方向

変更レジスタ

すべて

解　説

現在のジョイスティックまたはカーソルキーの状態をAレジスタに返します。値はBASICのSTICK関数と同じです。

GTTRIG(00D8H / MAIN)

機　能

トリガボタンの状態を読み出します。

コール手順

A　　トリガボタン番号

0　　スペースキー

1〜2　　トリガボタン

(マウスの場合は1〜2がポート1、2の左ボタン、3〜4が右ボタン)

戻り値

A　　トリガボタンまたはスペースバーの状態

0FFH　　押す

00H　　離す

変更レジスタ

AF

解　説

現在のトリガボタン、またはスペースキーの状態をAレジスタに返します。この値はトリガが押されていれば0FFH、押されていなければ0となります。

サンプルプログラム「JOYSTICK.MAC」を参照して下さい。

5.3 パドルの使用法

パドルの回路例は図6.22のとおりです。8端子にパルスを送ると単安定マルチバイブレータは、ある時間幅のパルスを発生します。この幅は可変抵抗の値によって10μs〜3msの範囲で変化しますので、パルス長を測定すれば可変抵抗の値、ひいてはその回転角度を知ることができます。

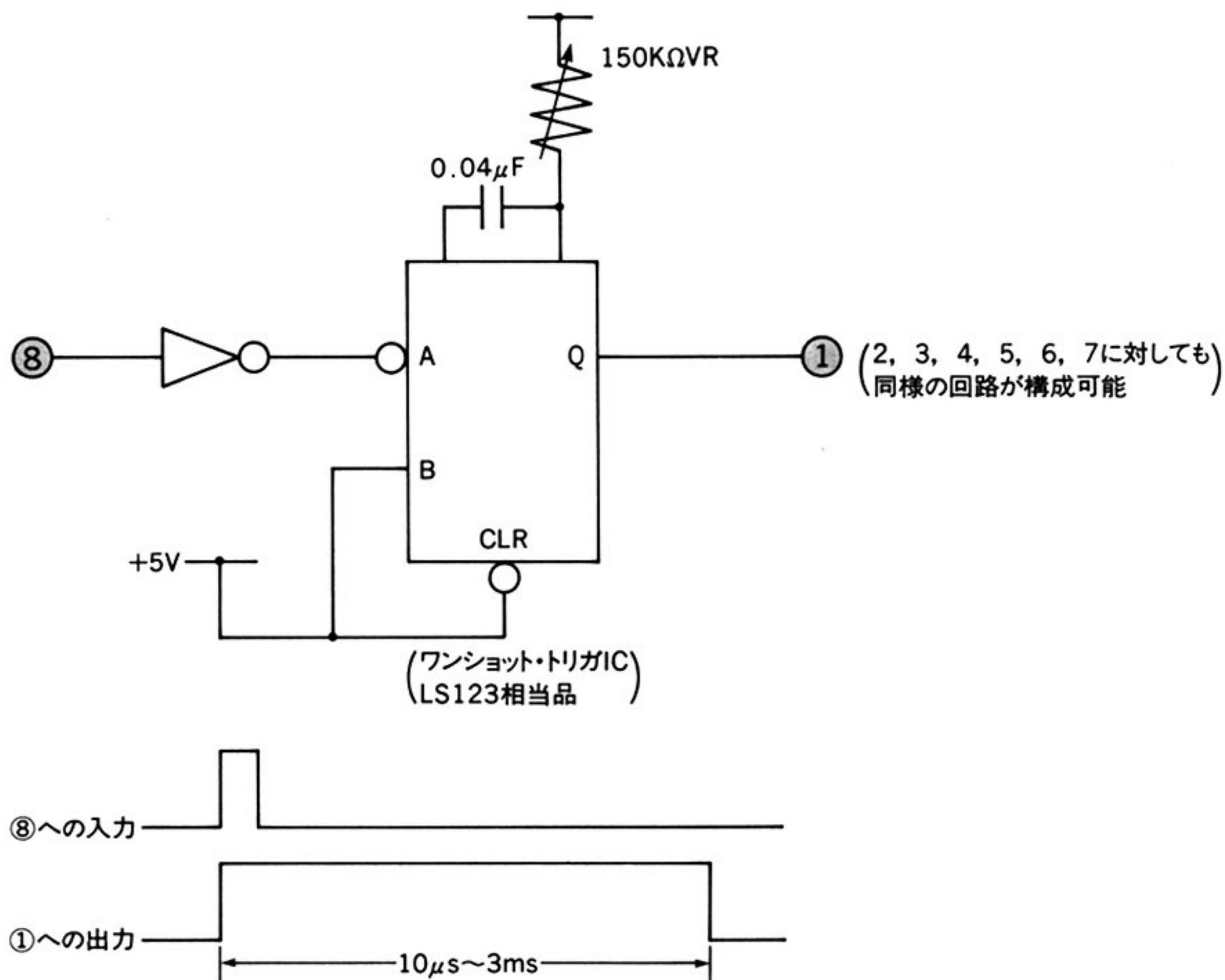



図6.22 パドル回路図

パドルをアクセスするためのBIOSルーチンを次に示します。

GTPDL(00DEH / MAIN)

機 能

パドル情報の状態を読み出します。

コール手順

A　　パドル番号（1〜12）

戻り値

A　　パドルの回転角（0〜255）

変更レジスタ

すべて

解　説

Aレジスタで指定したパドルの状態を調べ、結果をAレジスタに返します。

5.4 タッチパネル、ライトペン、マウス、トラックボールの使用法

タッチパネル、ライトペン、マウス、トラックボール（キャット）はすべて同一のBIOSを用いてアクセスすることができます。その使い方を以下に示します。

GTPAD(00DBH / MAIN)

機　能

各種入出力装置の状態を読み出します。

コール手順

A　　装置ID（0〜19）

戻り値

A　　目的の情報

変更レジスタ

すべて

解　説

Aレジスタで指定する値によって、表6.7のような各種の情報を得ることができます。これはBASICのPAD関数とまったく同じです。表中「XXX1」は、汎用入出力インターフェイス#1に接続した「XXX」の装置であることを意味します。同様に「XXX2」は汎用入出力インターフェイス#2に接続したものです。
サンプルプログラム「PAD.MAC」、「MOUSE.MAC」を参照して下さい。

表6.7　GTPADのBIOSで得られる情報

装置ID	指定される装置	返ってくる情報
0	タッチパネル1	パネル面に触っていれば0FFH、いなければ00H
1		X座標(0〜255)
2		Y座標(0〜255)
3		ボタンが押されていれば0FFH、いなければ00H
4 5 6 7	タッチパネル2	同　上
8	ライトペン*1	0FFHであればデータは有効、00Hであれば無効
9		X座標(0〜255)
10		Y座標(0〜255)
11		スイッチが押されていれば0FFH、いなければ00H
12	マウス1、または トラックボール1 *2 *3 *4	つねに0FFHを返す(入力の要求に使用)
13		X座標(0〜255)
14		Y座標(0〜255)
15		つねに00Hを返す(無意味)
16 17 18 19	マウス2、または トラックボール2	同　上

*1 ライトペンの座標(A＝9、10)とスイッチ(A＝11)の情報は、A＝8としてBIOSをコールしたとき同時に読み込まれますが、その結果が0FFHのときにのみ、他の値が有効となります。A＝8としてBIOSをコールした結果が00Hの場合は、その後に得られる座標値やスイッチの状態は意味を持ちません。
*2 マウスとトラックボールは自動的に判別します。
*3 マウスやトラックボールの座標値を求める場合、まず入力要求のコール(A＝12またはA＝16)を行い、その後に実際に座標値を求めるコールを実行する、という手順をふみます。このとき2つのコールの時間間隔はできる限り小さくしなければなりません。入力要求から座標の入力までに必要以上に時間をかけると、得られたデータは保証できません。
*4 マウスまたはトラックボールについているトリガボタンの状態を得るためには、GTTRIG(00D8H/MAIN)を使って下さい。

6章
CLOCKとバッテリバックアップメモリ

MSX2 は CLOCK-IC を用いて時計機能を実現しています。この IC はバッテリバックアップされており、MSX2 本体の電源を切っても動作を続けるようになっています。また、そのために内蔵されている RAM を、MSX2 では CLOCK 機能のほかにパスワードの設定やスタート時のスクリーンモードの自動設定などに利用しています。

6.1 CLOCK-IC の機能

この IC の機能は以下の 3 つに分けられます。

1. CLOCK 機能

■「年、月、日、曜日、時、分、秒」の設定／読み出しができます。

■時刻の表現は、24 時間計／12 時間計の切り換えができます。

■月の更新は、大の月と小の月を考慮します（4 年に一度の閏年も判別します)。

2. アラーム機能

■アラーム時刻を設定しておくと、CLOCK がその値に一致した時点で信号を発生します。

■アラーム時刻は「XX 日 XX 時 XX 分」の単位で設定できます。

3. バッテリバックアップメモリ機能

■ 26 個の 4 ビットメモリを持ち、各種の情報をバッテリバックアップすることができます。

■MSX2 では、このメモリに以下のようなデータを記憶させています。

1. CRT 表示の上下左右の調整値
2. SCREEN、WIDTH、COLOR の初期設定値
3. BEEP の音色と音量
4. タイトル画面の色
5. 国別コード
6. パスワード
7. BASIC のプロンプト
8. タイトル文字

（6～8 はどれかひとつが有効）

6.2 CLOCK-IC の構造

CLOCK-IC の内部は、図 6.24 に示すように 4 つのブロックに分かれています。各ブロックは 13 個の 4 ビットレジスタで構成され、それぞれのブロック内のレジスタは 0～12 のアドレスで指定します。またブロックの選択や機能コントロール用に 3 個の 4 ビットレジスタを持ち、これを 13～15 のアドレスで指定します。

各ブロックに含まれるレジスタ（# 0～# 12）と、MODE レジスタ（# 13）は書き込みも読み出しも可能です。TEST レジスタ（# 14）と RESET レジスタ（# 15）は書き込み専用で読み出しはできません。

BLOCK0 (CLOCK)
BLOCK1 (ALARM)
BLOCK2 (RAM-1)
BLOCK3 (RAM-2)
0 秒(1 の位)
1 秒(10 の位)
12 年(10 の位)
任意データ
任意データ
4ビット
13 MODE
14 TEST
15 RESET
書き込みのみ
4ビット

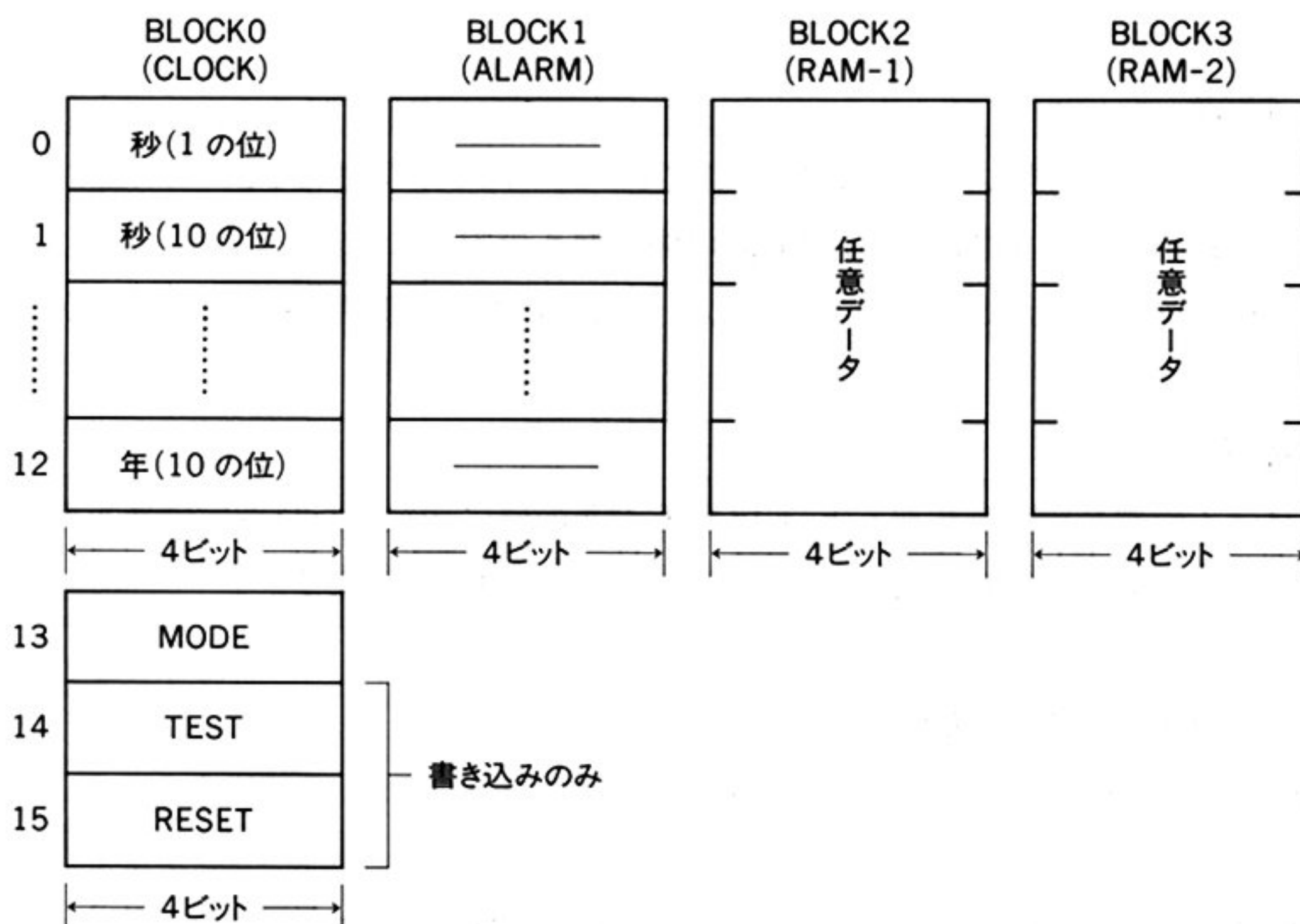



図 6.23 CLOCK-IC の構造

6.3 MODEレジスタの機能

MODEレジスタ（#13）は以下の3つの機能を持ちます。

1. ブロックの選択

#0〜#12のレジスタの読み書きを行う場合は、使用できるブロックを選択してから目的のアドレスをアクセスします。ブロックの選択にはMODEレジスタの下位2ビットを使用します。#13〜#15のレジスタは、どのブロックが選択されている場合でもアクセス可能です。

2. アラーム出力のON/OFF

MODEレジスタのビット2でアラーム出力のON/OFFを行います。ただし、MSX2ではアラームに関する仕様はオプションですから、このビットを書き換えても一般には何も起こりません。

3. CLOCKカウントの停止

MODEレジスタのビット3を「0」にすると、秒以降のカウントを停止し（秒より前の分周段は止まらない）、時計機能をストップさせます。ビット3を「1」にするとカウントを再開します。

3	2	1	0	
TE	AE	M1	M0	MODEレジスタ(#13)

M1 M0:
- 00 ブロック0を選択
- 01 ブロック1を選択
- 10 ブロック2を選択
- 11 ブロック3を選択

AE:
- 0 アラーム出力OFF
- 1 アラーム出力ON

TE:
- 0 CLOCKカウントの停止（秒以上）
- 1 CLOCKカウントの開始

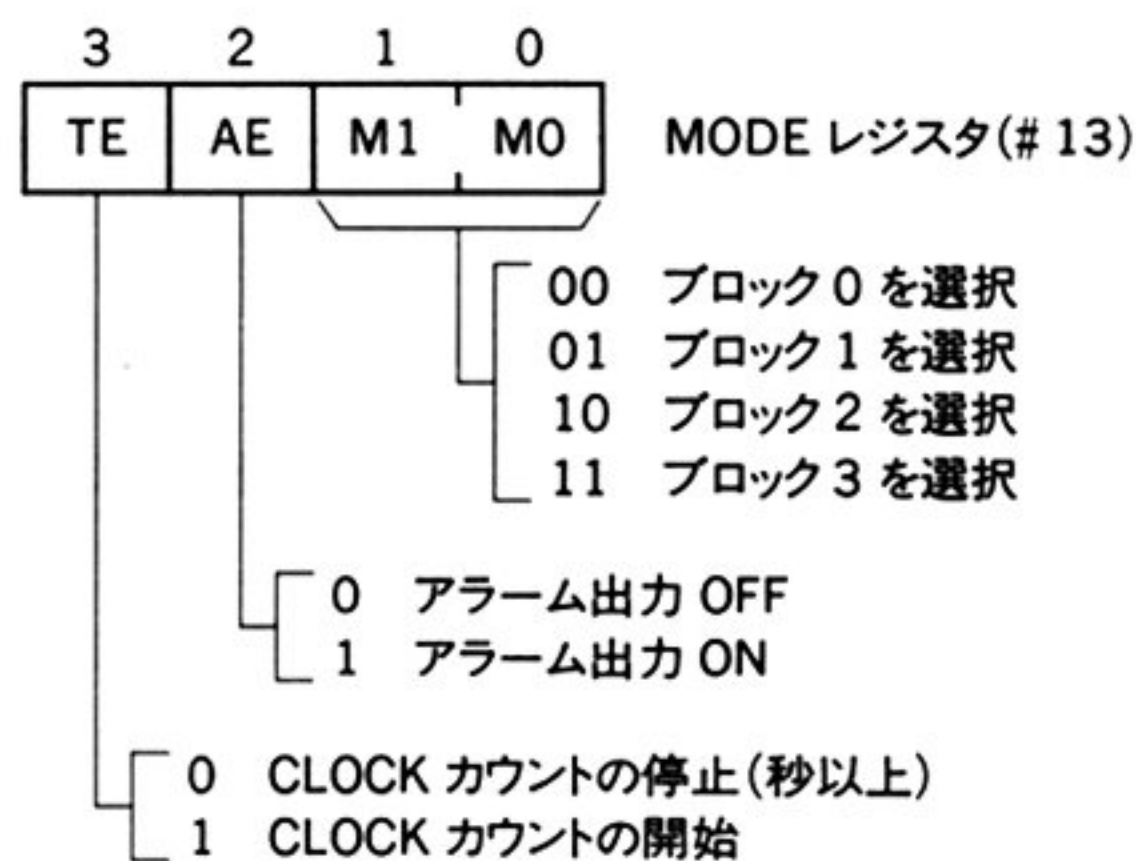



図6.24　MODEレジスタの機能

6.4 TESTレジスタの機能

TESTレジスタ(#14)は上位のカウンタを素早くカウントアップさせ、時刻や日付の繰り上がり動作を確認するために用います。このレジスタの各ビットに「1」を立てると、それぞれ日、時、分、秒のカウンタに直接214 (=16384) [Hz]のパルスが入力されます。

3	2	1	0	
T3	T2	T1	T0	TESTレジスタ(#14)
「日」	「時」	「分」	「秒」	………パルスが入力される位置

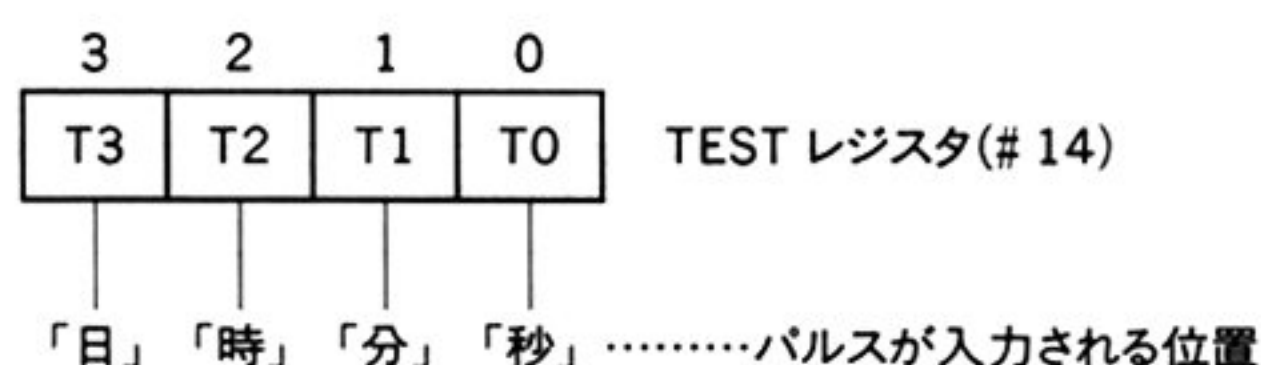



図6.25 TESTレジスタの機能

6.5 RESETレジスタの機能

RESETレジスタ (#15) には以下の機能があります。

1. アラームのリセット

ビット0を「1」にすると、すべてのアラームレジスタを0にリセットします。

2. 秒合わせ

ビット1を「1」にすると、秒以前の分周段をリセットします。この機能は1秒の始まりを正確に合わせる場合に用います。

3. クロックパルスのON/OFF

ビット2を「0」にすると、16Hzのクロックパルス出力をONに、ビット3を「0」にすると1Hzのクロックパルス出力をONにします。なお、両者ともMSX2ではサポートしていません。

3	2	1	0	
C1	C16	CR	AR	RESET レジスタ(# 15)

- AR:「1」で、すべてのアラームレジスタをリセット
- CR:「1」で、秒より前の分周段をリセット
- C16:「0」で、16[Hz]]クロックパルス ON
- C1:「0」で、1[Hz]]クロックパルス ON

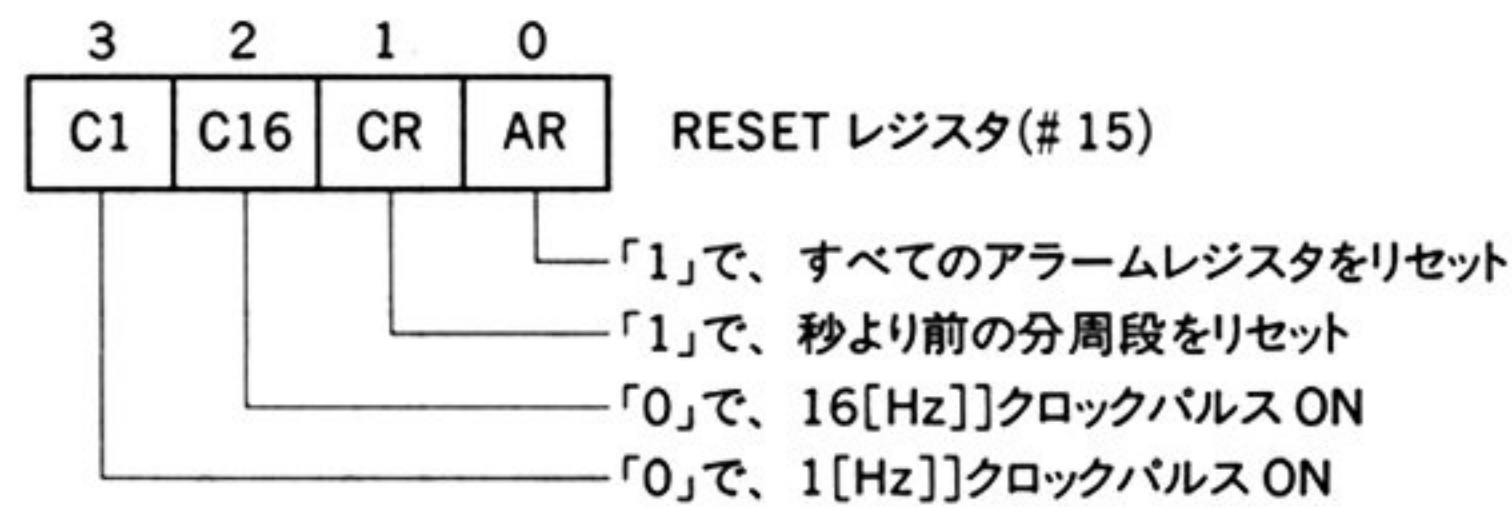



図 6.26　RESET レジスタの機能

6.6 クロックおよびアラームの設定

1. 日付と時刻の設定

クロックの設定にはブロック 0 を使用します。MODE レジスタでブロック 0 を選択し、目的とするレジスタにデータを書き込めば日付や時刻が設定されます。また、そのレジスタの内容を読み出せば現在の時間を知ることができます。レジスタの意味とそのアドレスについては図 6.27 を参照してください。

アラームの設定はブロック 1 を使用します。ただしアラーム時刻は日、時、分の単位しか指定できません。また、クロックがアラーム時刻に一致しても一般には何も起こりません。

クロックの中で、年は 2 桁 (レジスタ# 11〜# 12) で表されます。MSX BASIC では、この値にオフセット 80 を加えて西暦年数の下 2 桁を表すことにしています。例えば、レジスタ# 11 = 0、レジスタ# 12 = 0 と設定した後、BASIC の GET DATE 命令を用いて日付を読み出すと、「80 / XX / XX」のように、年数は 80 となります。

曜日は 0〜6 で表されます。これは単に日付とともに更新される 7 進カウンタで、実際の曜日と 0〜6 の数値の対応は決まっていません。

ブロック0(CLOCK)

		b3	b2	b1	b0
0	秒(1の位)	×	×	×	×
1	秒(10の位)	•	×	×	×
2	分(1の位)	×	×	×	×
3	分(10の位)	•	×	×	×
4	時(1の位)	×	×	×	×
5	時(10の位)	•	•	×	×
6	曜日	•	×	×	×
7	日(1の位)	×	×	×	×
8	日(10の位)	•	•	×	×
9	月(1の位)	×	×	×	×
10	月(10の位)	•	•	•	×
11	年(1の位)	×	×	×	×
12	年(10の位)	×	×	×	×

ブロック1(ALARM)

		b3	b2	b1	b0
0	——	•	•	•	•
1	——	•	•	•	•
2	分(1の位)	×	×	×	×
3	分(10の位)	•	×	×	×
4	時(1の位)	×	×	×	×
5	時(10の位)	•	•	×	×
6	曜日	•	×	×	×
7	日(1の位)	×	×	×	×
8	日(10の位)	•	•	×	×
9	——	•	•	•	•
10	12時or24時	•	•	•	×
11	閏年カウンタ	•	•	×	×
12	——	•	•	•	•

「•」で示したビットは常に「0」で、変更はできない。

図6.27 CLOCKとALARMの設定

2. 12時間計・24時間計の選択

時刻の表示は、昼の1時を「13時」と表す24時間計と、「午後1時」と表す12時間計のどちらかを選ぶことができます。この選択にはブロック1のレジスタ#10を使用します。図6.28に示したように、ビット0(b0)が「0」のとき12時間計、「1」のとき24時間計となります。

12時間計を選んだ場合は、図6.29のように10時間カウンタ(ブロック0、#5)のb1ビットによって午前・午後を表現します。

•	•	•	b0

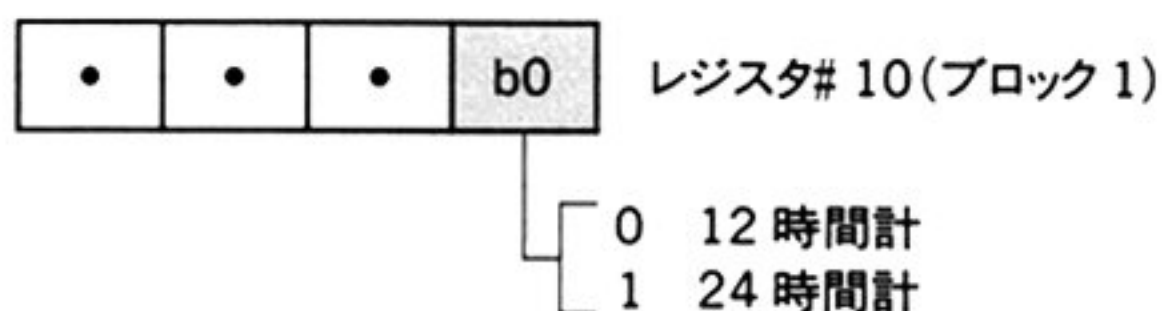
レジスタ# 10(ブロック1)

- 0 12時間計
- 1 24時間計

図6.28 12時間計・24時間計の選択

•	•	b1	×

レジスタ#5(ブロック0)

- 0 午前
- 1 午後

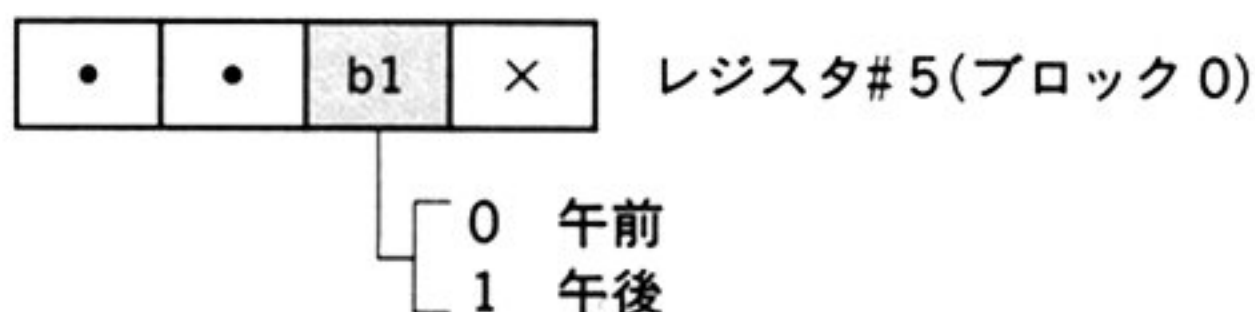



図6.29 12時間計の午前・午後フラグ

3. 閏年カウンタ

ブロック1のレジスタ#11は、年のカウントとともに更新される4進カウンタです。このレジスタの下位2ビットが00Hの場合は閏年とみなされ、2月を29日までカウントします。

MSX BASICで「SET DATE」の命令を実行すると、与えた年を4で割った余りがこのレジスタに設定されます。たとえば「80/XX/XX」という日付を指定すると閏年カウンタは0になりますが、これは西暦1980年が閏年であるという事実に一致するわけです。

•	•	b1	b0

レジスタ# 11(ブロック1)

ともに「0」の場合は閏年

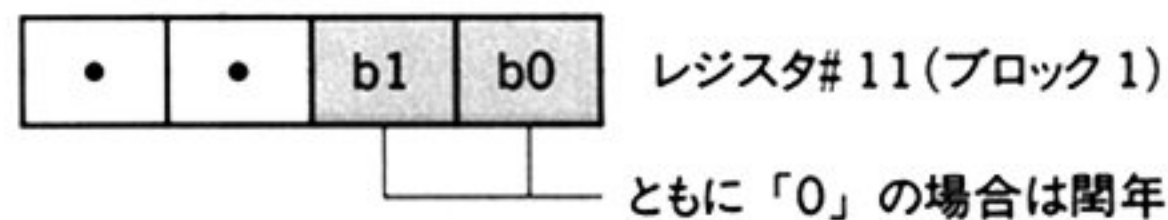



図6.30　閏年の判定

6.7 バッテリバックアップメモリの内容

CLOCK-ICのブロック2とブロック3は、それぞれ4ビット×13のバッテリバックアップされたメモリブロックとして用いられ、MSX2ではこの部分を以下のような用途に使っています。

1. ブロック2の内容

<table>
<tr><th></th><th>b3</th><th>b2</th><th>b1</th><th>b0</th></tr>
<tr><td>0</td><td colspan="4">ID</td></tr>
<tr><td>1</td><td colspan="4">Adjust X(−8〜+7)</td></tr>
<tr><td>2</td><td colspan="4">Adjust Y(−8〜+7)</td></tr>
<tr><td>3</td><td>——</td><td>——</td><td>Interlace Mode</td><td>Screen Mode</td></tr>
<tr><td>4</td><td colspan="4">WITDHの値(Lo)</td></tr>
<tr><td>5</td><td colspan="4">WITDHの値(Hi)</td></tr>
<tr><td>6</td><td colspan="4">前景色</td></tr>
<tr><td>7</td><td colspan="4">背景色</td></tr>
<tr><td>8</td><td colspan="4">周辺色</td></tr>
<tr><td>9</td><td>Cassette Speed</td><td>Printer Mode</td><td>Key Click</td><td>Key ON/OFF</td></tr>
<tr><td>10</td><td colspan="2">BEEP音色</td><td colspan="2">BEEP音量</td></tr>
<tr><td>11</td><td>——</td><td>——</td><td colspan="2">タイトルカラー</td></tr>
<tr><td>12</td><td colspan="4">国別コード</td></tr>
</table>

図6.31　ブロック2の内容

2. ブロック3の内容

ブロック3は、ID値(レジスタ#0)の内容により、3とおりの機能を持っています。図6.32にその機能を示します。

ID = 0:初期画面にタイトル(6文字以内)を表示する

0	0
1 2	Lo 1 Hi 1 ― タイトルの1番目の文字
⋮	⋮
11 12	Lo 6 Hi 6 ― タイトルの6番目の文字

(1〜12:6文字)

ID = 1:パスワードの設定

0	1
1 2 3	Usage ID = 1 Usage ID = 2 Usage ID = 3
4 5 6 7	パスワード パスワード パスワード パスワード ― パスワードデータを4ビット×4に圧縮して格納する
8	Keyカートリッジ存在フラグ
9 10 11 12	Keyカートリッジの値 Keyカートリッジの値 Keyカートリッジの値 Keyカートリッジの値

ID = 2:BASICのプロンプト設定

0	2
1 2	Lo 1 Hi 1 ― プロンプトの1番目の文字
⋮	⋮
11 12	Lo 6 Hi 6 ― プロンプトの6番目の文字

(1〜12:6文字)

図6.32 ブロック3の内容

6.8 CLOCK-ICのアクセス

クロックおよびバッテリバックアップメモリをアクセスするために、下記のようなBIOSルーチンが用意されています。このルーチンはSUB ROMにあるため、インタースロットコールを用いて呼び出します。

REDCLK(01F5H / SUB)

機　能

CLOCK-ICのデータを読み出します。

コール手順

C　　CLOCK-ICのアドレス（図6.33参照）

戻り値

A　　読み出したデータ（下位4ビットのみ有効）

変更レジスタ

すべて

解　説

Cレジスタで指定したアドレスのCLOCK-ICレジスタを読み出し、Aレジスタに格納します。アドレス指定には図6.33のようにブロック選択情報も含めているため、最初にMODEレジスタを設定し、次に目的のレジスタを読み出す、という手順をふむ必要はありません。

Cレジスタ	•	•	M1	M0	A3	A2	A1	A0
			選択するブロック		レジスタのアドレス			

図6.33　CLOCK-ICのレジスタ指定法

WRTCLK(01F9H / SUB)

機　能

CLOCK-IC へデータを書き込みます。

コール手順

C　　CLOCK-IC のアドレス（図 6.33 参照）
A　　書き込むデータ（下位 4 ビットのみ有効）

戻り値

なし

変更レジスタ

すべて

解　説

C レジスタで指定したアドレスの CLOCK-IC レジスタに、A レジスタの内容を書き込みます。アドレスは REDCLK と同様に、図 6.33 のようなフォーマットで指定します。

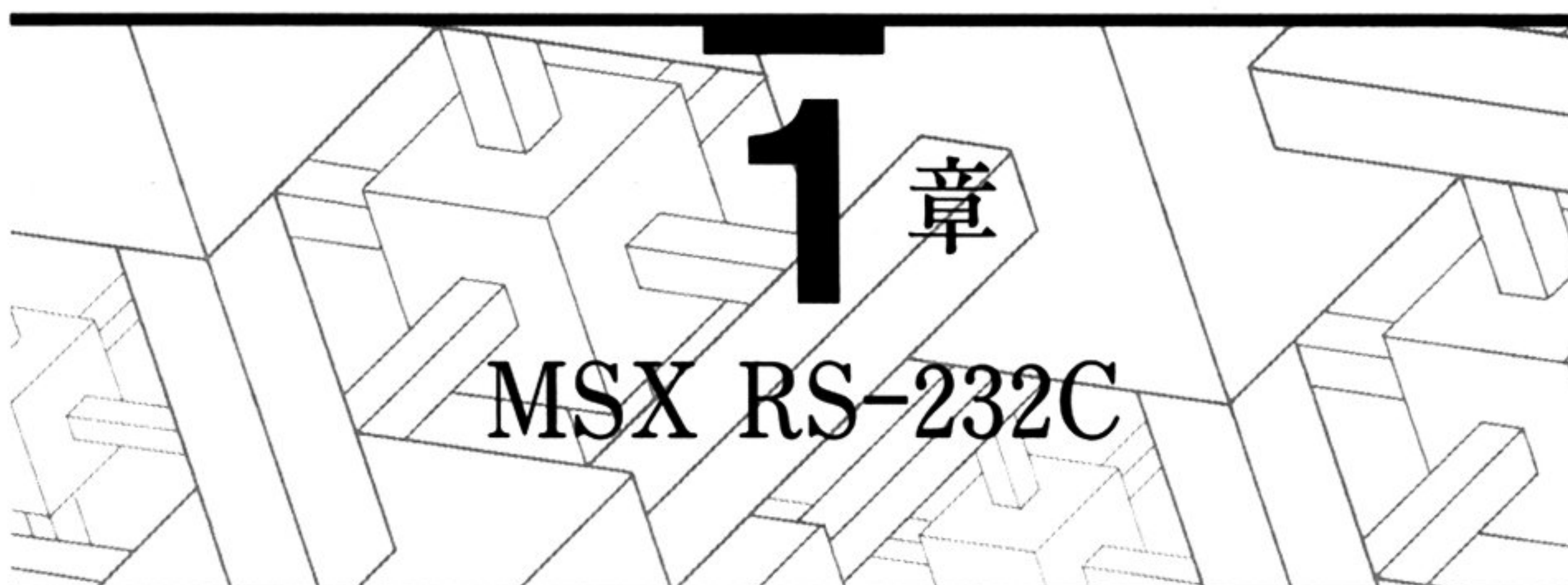
サンプルプログラム「PROMPT.MAC」を参照して下さい。

第7部

オプションの周辺装置のアクセス

MSXではオプションとして、さまざまな周辺装置が用意されています。それらの装置は、ハードウェアとシステムソフトウェアの仕様を定めることによって、既存のアプリケーションソフトウェアやユーザープログラムから使うことができます。第7部では、これらの周辺装置のハードウェア仕様とソフトウェア仕様を説明します。

オプションの周辺装置としては、MSX RS-232C、MSX MODEM、MSX-MUSIC、MSX-AUDIO、MSX-JE、24ドット漢字プリンタなどがあります。

1章
MSX RS-232C

1.1 ハードウェア

MSX RS-232C には、シングルチャンネルタイプとマルチチャンネルタイプの 2 つの仕様があります。それぞれについて、以下で解説します。

1.1.1 シングルチャンネルバージョン

1. 使用するLSI

i8251	通信インターフェイス
i8253	プログラマブルインターバルタイマ
ROM	システムソフトウェア用（8K バイト）

2. ポートアドレス

表7.1 シングルチャンネルタイプのポートアドレス

I/Oアドレス	Read/Write	意　味
80H	R/W	8251 データポート
81H	R/W	8251 コマンド/ステータスポート
82H	R	CTS、Timer/Counter2、RI、CD用ステータスセンス
82H	W	割り込みマスクレジスタ
83H		システム予約
84H	R/W	8253 カウンタ 0
85H	R/W	8253 カウンタ 1
86H	R/W	8253 カウンタ 2
87H	W	8253 モードレジスタ

3. 82H番地ビット割り当て

82H Read　　システムステータスの獲得

表7.2 シングルチャンネルタイプのシステムステータス

データビット	意　味	
D7	CTS (Clear To Send)	
	0	CTS Asserted
	1	CTS Negated
D6	タイマ/カウンタ出力2 (8253より)	
D5	システム予約	
D4	システム予約	
D3	システム予約	
D2	システム予約	
D1	RI (Ring Indicator)*	
	0	RI Asserted
	1	RI Negated
D0	CD (Carrier Detect)*	
	0	CD Asserted
	1	CD Negated

注　意　*印の信号はオプション。どちらか一方をインプリメントする場合、必ずCD信号にする。

82H Write　割り込みマスクレジスタ

表 7.3　シングルチャンネルタイプの割り込みマスクレジスタ

データビット	意　味		
D7	システム予約		
D6	システム予約		
D5	システム予約		
D4	システム予約		
D3	Timer Interrupt from i8253 channel2*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D2	Sync character detect / Break detect*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D1	Transmit Data Ready（TxReady）*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D0	Receive Data Ready（RxReady）		
		1	割り込み不許可（初期値）
		0	割り込み許可

注　意　*印の信号はオプション。最小構成では、割り込み信号は RxReady のみ。

4. 8253を使用した8251へのボーレートクロック発生

1. 水晶発振器　発振周波数 1.8432MHz

表7.4 シングルチャンネルタイプのボーレートとスケールファクタ

ボーレート	スケールファクタと誤差		
50	2304		
75	1536		
110	1047	110.0287	+0.03%
150	768		
300	384		
600	192		
1200	96		
1800	64		
2000	58	1986.2	−0.7%
2400	48		
3600	32		
4800	24		
7200	16		
9600	12		
19200	6		

2. 使用するカウンタチャンネル

CH0	Rx Baud rate clock
CH1	Tx Baud rate clock
CH2	Used by Application (Interrupt generated optionally)

5. DSUB25 コネクタのピン配列

表 7.5　DSUB25 コネクタのピン配列

ピン番号	信号名	ピン番号	信号名
1	Frame Ground	14	
2	Transmit Data	15	
3	Receive Data	16	
4	Request To Send	17	
5	Clear To Send	18	
6	Data Set Ready	19	
7	Signal Ground	20	Data Terminal Ready
8	Carrier Detect	21	
9		22	RING Indicator
10		23	
11		24	
12		25	
13			

1.1.2 マルチチャンネルバージョン

1. 使用する LSI

i8251　通信インターフェイス
i8253　プログラマブルインターバルタイマ
ROM　システムソフトウェア用 (8K バイト)
RAM　受信バッファ、ワークエリア用 (2K バイト)

2. ROM アドレス

RS-232C のシステムソフトウェアは 4000H～5FFFH に配置します。ROM は以下のモジュールを含みます。

1. RS-232C ドライバ
2. RS-232C 拡張 BASIC
3. RS-232C 拡張 BIOS

マルチチャンネルバージョンでサポートしているチャンネル数は1つのカートリッジにつき、最大4チャンネル（1台のシステムにつき、最大4チャンネル）です。

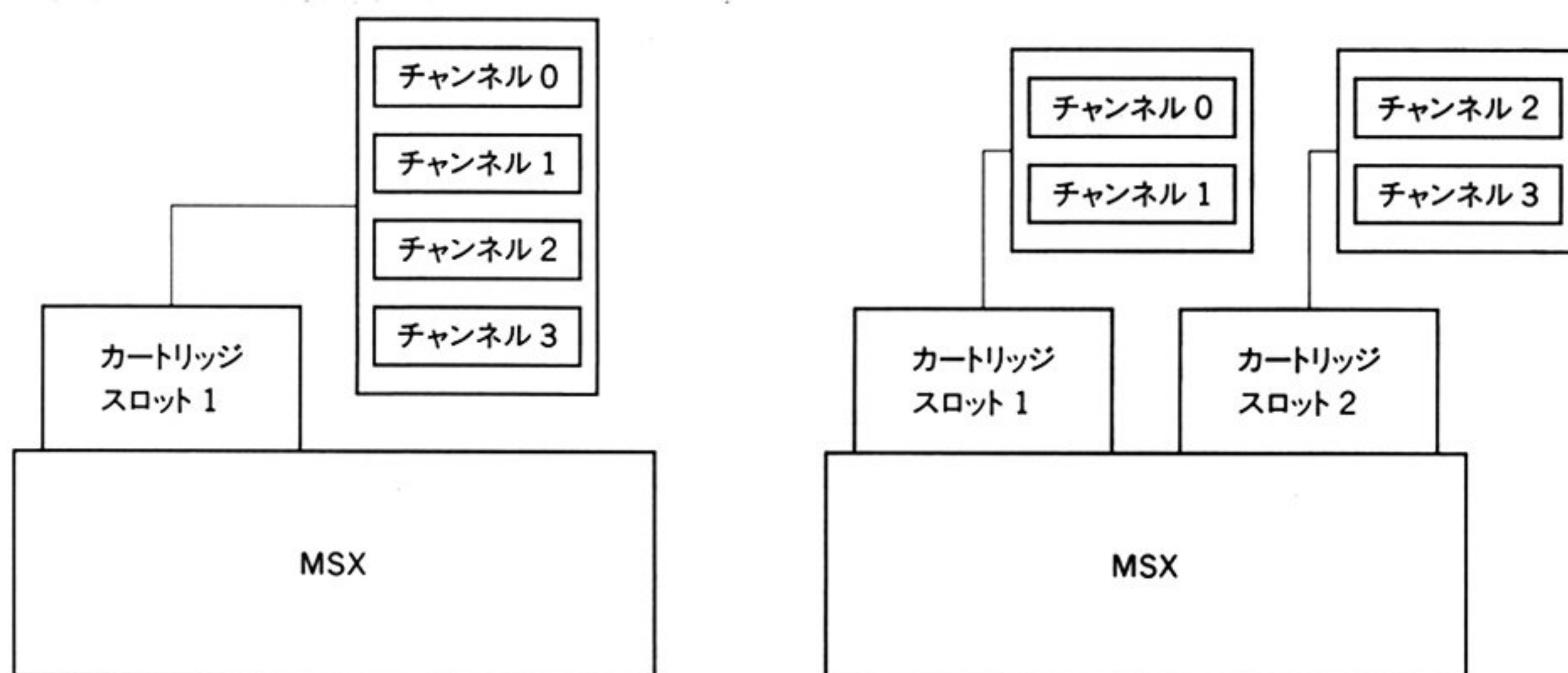



図7.1　マルチチャンネルRS-232Cカートリッジの実装例

3. RAMアドレス

RAMは6000H〜67FFHに配置します。また、同じRAMがA000H〜A7FFHにも見えるようにイメージを出します。したがって、6000Hに1を書き込むと、A000Hも1になります。システムソフトウェアから両方のアドレスのRAMにアクセスできるように回路を設計しなければなりません。これは、RS-232Cカートリッジの実行速度をなるべく高速にするためです。

RAMは以下の目的で使われます。

1. 1チャンネルにつき、128バイトのキャラクタとそれぞれのキャラクタについてのエラー情報128バイト、合計256バイトの受信用バッファ
2. フラグと変数の保存
3. 通信パラメータの保存

4. ポートアドレス

マルチチャンネルRS-232Cをサポートするためには、アドレスの衝突を防ぐためにすべてのポートはメモリ上に置かなければなりません。しかし、シングルチャンネルRS-232CのI/Oポートを直接アクセスしているアプリケーションソフトウェアを救済するために、最初の1チャンネルはI/Oポートに接続して下さい。また、この仕様については、システムソフトウェアでコントロールが可能でなければなりません。

表 7.6　マルチチャンネルタイプのメモリアドレス

メモリアドレス	I/O アドレス	Read / Write	意　味
BFF8H	80H	R/W	8251 データポート
BFF9H	81H	R/W	8251 コマンド / ステータスポート
BFFAH	82H	R	CTS、Timer / Counter2、RI、CD 用ステータスセンス
BFFAH	82H	W	割り込みマスクレジスタ
BFFBH	83H		システム予約
BFFCH	84H	R/W	8253 カウンタ 0
BFFDH	85H	R/W	8253 カウンタ 1
BFFEH	86H	R/W	8253 カウンタ 2
BFFFH	87H	W	8253 モードレジスタ

これはチャンネル 0 のメモリアドレスです。各チャンネルの最下位アドレスは以下の通りです。

- チャンネル 1　　BFF0H
- チャンネル 2　　BFE8H
- チャンネル 3　　BFE0H

チャンネルは連続し、0 から始まらなければなりません。

表 7.7　チャンネル配置の例

良い例	悪い例
チャンネル 0 のみ	チャンネル 0 と 2
チャンネル 0 と 1	チャンネル 0 と 3
チャンネル 0、1、2	チャンネル 1 のみ
チャンネル 0、1、2、3	チャンネル 1 と 2
	チャンネル 1、2、3
	チャンネル 3 のみ
	など

チャンネル 0 は特別なチャンネルで、I/O ポートのイネーブルビット（BFFAH のビット 4）を持っています。そのビットが 1 の時、チャンネル 0 は I/O ポートを通してアクセスできます。これは、シングルチャンネルの RS-232C インターフェイスとの互換性を保つためです。

システムソフトウェアはまず、I/O に RS-232C インターフェイスが接続されているかどうかを調べます。接続されていなければ、BFFAH のビット 4 に 1 をセットし、チャンネル 0 を I/O

に接続します。これにより、I/Oポートを直接アクセスしているアプリケーションソフトウェアでもマルチチャンネルのRS-232Cで動作可能になります。

I/Oポートの82Hは、ビット4以外はBFFAHと同じです。アプリケーションが誤ってビットを反転し、RS-232Cがアクセス不可能にならないよう、このビットはI/Oポートからはアクセスできません。

メモリのBFFBH（I/Oの83H）番地は、システム用に予約されています。

5. アドレスマップ

RAMアドレス

アドレス	内容
BFFFH	
	チャンネル0の I/Oエリア
BFF8H	
	チャンネル1の I/Oエリア
BFF0H	
	チャンネル2の I/Oエリア
BFE8H	
	チャンネル3の I/Oエリア
BFE0H	
	未使用
A800H	
	RS-232C用のRAM (イメージ)
A000H	
	未使用
8000H	
	未使用
6800H	
	RS-232C用のRAM
6000H	
	システムソフトウェア用ROM (最小8Kバイト)
4000H	

I/Oアドレス

アドレス	内容
87H	
	チャンネル0の I/Oエリア (接続されている場合)
80H	

図7.2　アドレスマップ

6. BFFAH(メモリ)、82H(I/O)番地ビット割り当て

Read　システムステータスの獲得

表7.8　マルチチャンネルタイプのシステムステータス

データビット	意　味		
D7	CTS (Clear To Send)		
		0	CTS Asserted
		1	CTS Negated
D6	タイマ/カウンタ出力2 (8253より)		
D5	システム予約		
D4	システム予約		
D3	システム予約		
D2	システム予約		
D1	RI (Ring Indicator)*		
		0	RI Asserted
		1	RI Negated
D0	CD (Carrier Detect)*		
		0	CD Asserted
		1	CD Negated

注　意　*印の信号はオプション。どちらか一方をインプリメントする場合、必ずCD信号にする。

Write　割り込みマスクレジスタ

表7.9　マルチチャンネルタイプの割り込みマスクレジスタ

データビット	意　味		
D7	システム予約（必ず1）		
D6	システム予約（必ず1）		
D5	システム予約（必ず1）		
D4	チャンネル0のI/Oポートのアクセス（BFFAHのみ）		
		1	I/Oポートの接続を許可
		0	I/Oポートの接続を禁止（初期値）
D3	Timer Interrupt from i8253 channel 2*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D2	Sync character detect / Break detect*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D1	Transmit Data Ready（Tx Ready）*		
		1	割り込み不許可（初期値）
		0	割り込み許可
D0	Receive Data Ready（Rx Ready）		
		1	割り込み不許可（初期値）
		0	割り込み許可

注　意　*の信号はオプション。すなわち最小構成では、割り込み信号はRxReadyのみ。

7. 8253を使用した8251へのボーレートクロック発生

1. 水晶発振器　発振周波数 1.8432MHz

表7.10　マルチチャンネルタイプのボーレートとスケールファクタ

ボーレート	スケールファクターと誤差		
50	2304		
75	1536		
110	1047	110.0287	+0.03%
150	768		
300	384		
600	192		
1200	96		
1800	64		
2000	58	1986.2	−0.7%
2400	48		
3600	32		
4800	24		
7200	16		
9600	12		
19200	6		

2. 使用するカウンタチャンネル

CH0	Rx Baud rate clock
CH1	Tx Baud rate clock
CH2	Used by Application (Interrupt generated optionally)

8. DSUB25 コネクタのピン配列

表7.11 DSUB25 コネクタのピン配列

ピン番号	信号名	ピン番号	信号名
1	Frame Ground	14	
2	Transmit Data	15	
3	Receive Data	16	
4	Request To Send	17	
5	Clear To Send	18	
6	Data Set Ready	19	
7	Signal Ground	20	Data Terminal Ready
8	Carrier Detect	21	
9		22	RING Indicator
10		23	
11		24	
12		25	
13			

1.2 拡張 BASIC

1.2.1 概要

MSX RS-232C には、各機能を簡単に使用できるように、MSX RS-232C 拡張 BASIC が用意されています。コマンド、ステートメント、関数は MSX BASIC、MSX Disk BASIC と同じですが、書式や指定方法の違いにより、RS-232C 通信が行なえるように機能が拡張されています。

使い方は、CALL COMINI のように拡張ステートメントの形式です。CALL は_（アンダーバー）で代用できます。

デバイス番号は、スロット番号の小さいスロットにさしこまれている通信用カートリッジの順に、0 から割り当てられます。モデムカートリッジなど、MSX RS-232C 以外の通信用カートリッジといっしょに使用された場合にも、同様にスロット順に 0 から割り当てられます。

通信用カートリッジには、RS-232C インターフェイス、MODEM カートリッジ、「MSX -SERIAL232」などがあります。

1.2.2 この章の表記法

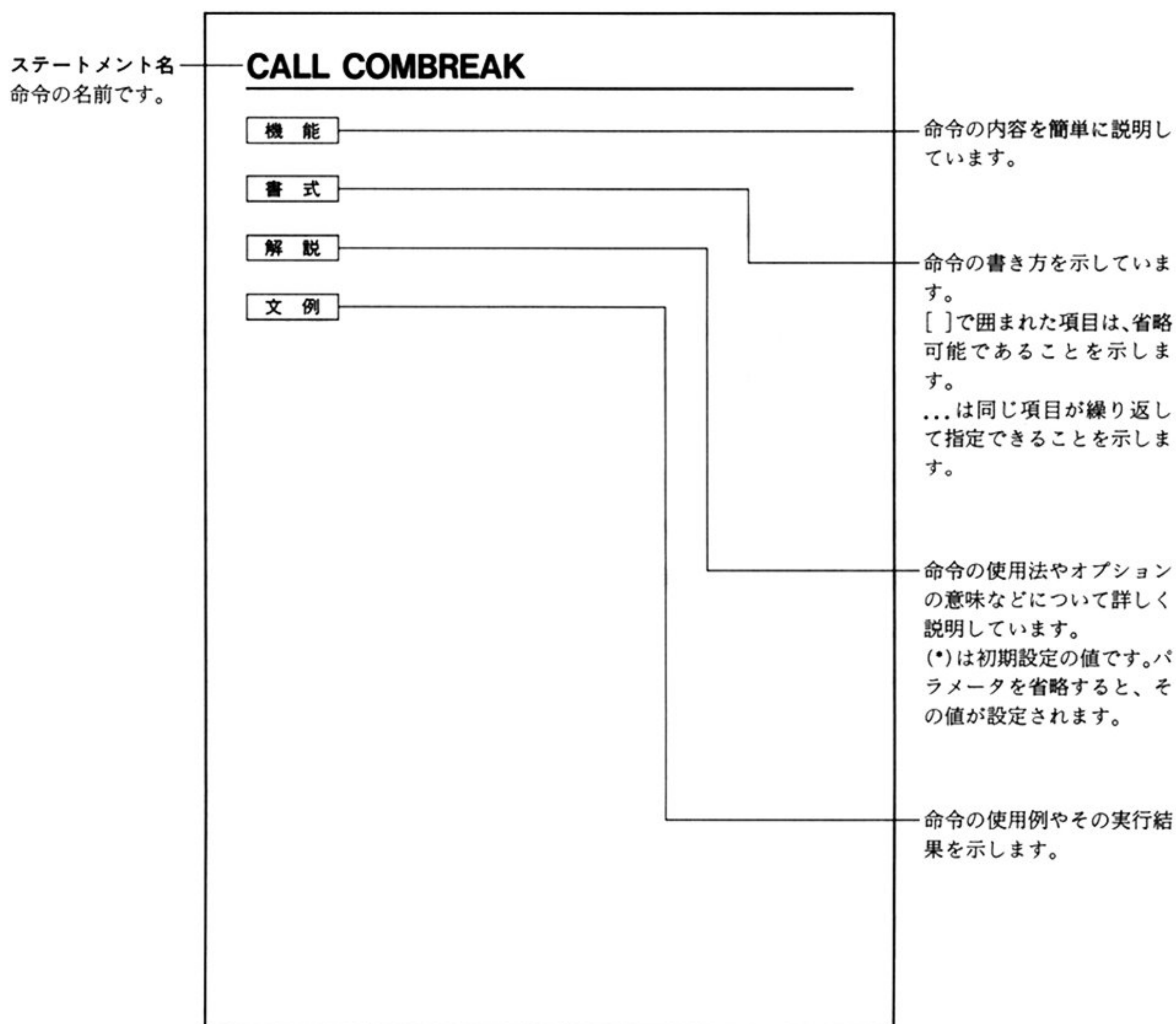
ステートメント名
命令の名前です。

CALL COMBREAK

機能
命令の内容を簡単に説明しています。

書式
命令の書き方を示しています。
[ ]で囲まれた項目は、省略可能であることを示します。
...は同じ項目が繰り返して指定できることを示します。

解説
命令の使用法やオプションの意味などについて詳しく説明しています。
(*)は初期設定の値です。パラメータを省略すると、その値が設定されます。

文例
命令の使用例やその実行結果を示します。

1.2.3 拡張 BASIC コマンド一覧

1. 拡張ステートメント(CALL 文と共に使用します)

コマンド名	機　能	ページ
COMBREAK	ブレーク信号を送信します	103
COMDTR	DTR (ER) 信号を ON/OFF します	104
COM GOSUB	RS-232C からの割り込み処理サブルーチンの開始行を指定します	104
COMHELP	COMINI のパラメータ指定方法をヘルプメッセージとして出力します	105
COMINI	通信機能の初期設定をします	106
COMOFF	RS-232C からの割り込みを禁止します	109
COMON	RS-232C からの割り込みを許可します	110
COMSTAT	RS-232C のステータスを求めます	111
COMSTOP	RS-232C からの割り込みを保留します	113
COMTERM	ターミナルモードにします	113

2. コマンド

コマンド名	機　能	ページ
LOAD	プログラムを受信します	117
MERGE	プログラムを受信してメモリ上のプログラムとマージします	117
RUN	プログラムを受信した後に実行を開始します	118
SAVE	プログラムを送信します	119

3. ステートメント

コマンド名	機　能	ページ
CLOSE	RS-232C 用ファイルをクローズします	120
INPUT #	数値や文字を入力して変数に代入します	120
LINE INPUT #	254 文字までの文字を受信して文字型変数に代入します	121
OPEN	RS-232C 用ファイルをオープンします	122
PRINT #	数値や文字を送信します	123
PRINT # USING	書式付き PRINT #です	125

4. 関数

コマンド名	機　能	ページ
EOF	EOF コード（&H1A）が受信されたかどうかを求めます	127
INPUT$	指定した数の文字を入力します	127
LOC	受信バッファ内の文字数を獲得します	128
LOF	通信用バッファの空き容量を獲得します	129

注　意

COMON、COMOFF、COMSTOP の説明にある、RS-232C からの割り込みとは、BASIC のプログラム実行に対する割り込みです。RS-232C ポート（ハードウェア）から発生する実際の割り込みではありません。

1.2.4 拡張 BASIC の解説

1. 拡張ステートメント

CALL COMBREAK

機　能

ブレーク信号を送信します。

書　式

CALL COMBREAK[(["デバイス番号：”],式)]

解　説

SD（送信データ）を強制的にブレーク状態にします。ブレーク状態とは回線が切断状態になることです。

デバイス番号

設定の対象となるカートリッジを 0（*）～9 の整数値で指定します。省略するときはコロン（：）まで省略します。

式

数値定数、変数、配列変数、式で指定します。指定できる範囲は 3 から 32767 までの値です。式の値は次のように計算します。

$$\text{値の式} = \frac{\text{送信速度} \times \text{ブレーク送信時間}}{1 + \text{キャラクタ長} + \text{ストップビット長}}$$

省略した場合は、10 になります。

文　例

CALL COMBREAK(,480)

もしくは

CALL COMBREAK("0:",480)

CALL COMINI において以下のように設定した場合にデバイス番号 0 に対して、ブレーク信号を 500m 秒送信します。

キャラクタ長	8 ビット
ストップビット長	1 ビット
送信速度	9600bps

CALL COMDTR

機 能

DTR (ER) 信号をON/OFFします。

書 式

CALL COMDTR(["デバイス番号:"],式)

解 説

デバイス番号

設定の対象となるカートリッジを0 (*) ~9の整数値で指定します。省略するときはコロン (:) まで省略します。

式

0と1の値で指定します。

0	DTR (ER) をOFF
1	DTR (ER) をON 0以外の数値を指定した場合にも、DTR (ER) はONになる

文 例

CALL COMDTR(,0)

もしくは

CALL COMDTR("0:",0)

デバイス番号0のDTR (ER) 信号をOFFにします。

CALL COM GOSUB

機 能

RS-232Cからの割り込み処理サブルーチンの開始行を指定します。

書 式

CALL COM(["デバイス番号:"],GOSUB 行番号)

解 説

RS-232Cから割り込みがかかると、指定したサブルーチンが実行されます。

CALL COMON文を使って割り込みを許可してからデータが受信されると、実行中の文の実行が終わってから、行番号で指定したサブルーチンが実行されます。サブルーチンはRETURN文で終了します。サブルーチンの実行終了後は、割り込みがかかった時点で、実行していたステートメントの次のステートメントに実行が戻ります。
割り込み処理サブルーチンの実行中は、自動的にCOMSTOPとなり、COMOFFしないかぎり、RETURN文によってCOMON状態となります。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。省略するときはコロン（：）まで省略します。

行番号

0から65529までの整数型定数で指定します。

文　例

CALL COM(,GOSUB 10000)
もしくは
CALL COM("0:",GOSUB 10000)
デバイス番号0から割り込みがかかると行番号10000にGOSUBします。

CALL COMHELP

機　能

COMINIのパラメータ指定方法がヘルプメッセージとして出力します。

書　式

CALL COMHELP[("デバイス番号：")]

解　説

次のようにメッセージを送信します。

```
Initialize statement options

CALL COMINI ("
<device # {0, 1, 2, . . .  9} > :
<character length {5, 6, 7, 8} >
<parity {E, O, I, N} >
```

```
<stop bits {1, 2, 3} >
<XON/XOFF {X, N} >
<CTS hand-shake {H, N} >
<auto LF on receive {A, N} >
<auto LF on transmit {A, N} >
<SI/SO {S, N} >"
, <recieve baud rate>
, <transmitter baud rate>
, <time out count>
Default:
CALL COMINI("0:8N1XHNNN"
        , 1200,1200,0)
```

CALL COMINI

機能

通信機能の初期設定をします。

書式

CALL COMINI[(["[デバイス番号:][キャラクタ長[パリティ[ストップビット[XON/OFF 制御[RS/CS 制御[受信 LF 挿入[送信 LF 削除[SI/SO 制御]]]]]]]]"][,[受信速度][,[送信速度][,[送信タイムアウト]]]])]

解説

通信機能の初期設定をします。指定する文字は大文字でも小文字でもかまいません。

デバイス番号

設定の対象となるカートリッジを0(*)～9の整数値で指定します。省略するときはコロン(:)まで省略します。

キャラクタ長

接続する機種に合わせて送信する1キャラクタのビット数を指定します。

5	5ビット
6	6ビット
7	7ビット
8 (*)	8ビット

パリティ

パリティチェックの方法を指定します。

E	偶数パリティ
O	奇数パリティ
I	受信時にパリティを無視する（キャラクタ長が5、6、7ビットのときに有効)。送信時にはパリティビットを0にする
N (*)	パリティを使用しない

ストップビット長

ストップビットのビット数を指定します。

1 (*)	1ビット
2	1.5ビット
3	2ビット

XON/OFF制御

XON/OFFによるフロー制御を行うかどうかを指定します。

X (*)	XON/OFF制御を行う
N	XON/OFF制御を行わない
	XONコード=&H11、XOFFコード=&H13

RS/CS制御

受信バッファの空き領域が少なくなったときに、RSをOFFして相手側からの送信を中断させ、バッファが空くと、RSをONして送信を再開させます。CSがOFFだと、相手への送信を中断して、CSがONになると送信を再開します。

H (*)	RS/CS制御を行う
N	RS/CS制御を行わない

受信LF挿入

CR（0DH）コードを受信したときに、CRコードとLF（0AH）コードに変換する制御です。

A	CRコードとLFコードに変換する
N (*)	CRコードとLFコードに変換しない

送信LF削除

CR（0DH）コードの次にLF（0AH）コードを続けて送信するとき、LFコードを削除して、CRコードのみを送信する制御です。

A	LFを削除する
N (*)	LFを削除しない

SI/SO 制御

キャラクタ長が7ビットのときに、0A1Hコードから0FEHコードを送受信するときの制御を指定します。SI (0EH) コードによりG0集合を呼びだし、SO (0FH) コードによりG1集合を呼び出します。*1
G0集合とは21Hコードから7EHコードまでの集合を、G1集合とは0A1Hコードから0FEHコードまでの集合を指します。

S	SI/SO 制御を行う
N (*)	SI/SO 制御を行わない

受信速度

データ受信速度を指定します。単位はビット/秒です。

50、75、110、300、600、1200(*)、1800、2000、2400、3600、4800、7200、9600、19200

送信速度

データ送信速度を指定します。指定できる速度および指定方法はデータ受信速度と同じです。ただし、送信速度と送信タイムアウトをカンマ(,)まで含めて省略すると、送信速度は受信速度と同じになります。

送信タイムアウト

データを送信するときに、

XOFFコードを受信している
CSがOFFのためデータが送信できない

などの場合の待ち時間を指定します。単位は約1秒です。タイムアウトした場合には送信を中止して、

Device I/O error

というエラーメッセージを表示します。省略した場合や、0を指定した場合は、タイムアウトしません。指定できるのは、0～255の整数値です。

*1 SI/SO制御を行うときは&H80～&H9Fのコードは送信しないで下さい。コントロールコード(&H00～&h1F)に変換されて送信されるため、正常な交信が行われなくなる場合があります。

文　例

CALL COMINI("0:8N1XHNNN",9600,,3)

この例は、RS-232C を以下のように設定します。

デバイス番号	0
キャラクタ長	8 ビット
パリティ	N
ストップビット長	1 ビット
XON / OFF 制御	X
RS / CS 制御	H
受信 LF 挿入	N
送信 LF 削除	N
SI / SO 制御	N
受信速度	9600bps
送信速度	9600bps
送信タイムアウト	3 秒

CALL COMINI("7E3NNAAS")

この例は、RS-232C を以下のように設定します。

デバイス番号	0
キャラクタ長	7 ビット
パリティ	E
ストップビット長	2 ビット
XON / OFF 制御	N
RS / CS 制御	N
受信 LF 挿入	A
送信 LF 削除	A
SI / SO 制御	S
受信速度	1200bps
送信速度	1200bps
送信タイムアウト	なし

CALL COMOFF

機　能

RS-232C からの割り込みを禁止します。

書 式

CALL COMOFF[("デバイス番号：")]

解 説

CALL COMOFF 文を実行後、RS-232C にデータが来ても、割り込みは発生しません。

デバイス番号

設定の対象となるカートリッジを 0(*)～9 の整数値で指定します。省略するときはカッコごと省略します。

文 例

CALL COMOFF
もしくは
CALL COMOFF("0:")
デバイス番号 0 を COMOFF します。

CALL COMON

機 能

RS-232C からの割り込みを許可します。

書 式

CALL COMON[("デバイス番号：")]

解 説

CALL COMON 文を実行後、RS-232C にデータが来ると COM GOSUB 文で指定してある行番号が実行されます。

デバイス番号

設定の対象となるカートリッジを 0(*)～9 の整数値で指定します。省略するときはカッコごと省略します。

文 例

CALL COMON

もしくは

CALL COMON("0:")

デバイス番号0をCOMONします。

CALL COMSTAT

機 能

RS-232Cのステータスを求めます。

書 式

CALL COMSTAT(["デバイス番号："],数値変数)

解 説

RS-232Cのステータスを数値変数に代入します。

ステータスは数値変数値の下位16ビットが1か0かで示します。

ビット0　受信キャリアの検出

(LSB)　0　キャリアを検出していない

1　キャリアを検出

ビット1　被呼表示（リングインジケータ）の検出

0　被呼表示を検出していない

1　被呼表示を検出

被呼表示がある間のみ、1になる

ビット2　ブレーク信号の検出

0　ブレーク信号を検出していない。

1　ブレーク信号を検出した。

前回のCOMSTAT実行後から今回までに、ブレーク信号を検出した場合、1になる。

ビット3　データセットレディ

0　DR（DSR）信号を検出していない。

1　DR（DSR）信号を検出。

ビット4　未使用（常に0）

ビット5　未使用（常に0）

ビット6　タイマ/カウンタ出力2
　0　タイマ/カウンタ出力2が偽である。
　1　タイマ/カウンタ出力2が真である。
ビット7　CS信号検出
　0　CS（CTS）信号を検出していない。
　1　CS（CTS）信号を検出。
ビット8　未使用（常に0）
ビット9　未使用（常に0）
ビット10　[CTRL]＋[STOP]キー検出
　0　押されていなかった
　1　押されていた
ビット11　パリティエラー
　0　エラーなし
　1　エラーあり
ビット12　オーバーランエラー
　0　エラーなし
　1　エラーあり
ビット13　フレーミングエラー
　0　エラーなし
　1　エラーあり
ビット14　送信タイムアウトエラー
　0　エラーなし
　1　エラーあり
ビット15　バッファオーバーフローエラー
(MSB)　0　エラーなし
　1　エラーあり

文　例

```
CALL COMSTAT(,F):PRINT BIN$(F)
```

もしくは

```
CALL COMSTAT("0:",F):PRINT BIN$(F)
```

デバイス番号0のステータスを数値変数Fに代入して、画面に2進数表示をします。

CALL COMSTOP

機 能

RS-232C からの割り込みを保留します。

書 式

CALL COMSTOP[("デバイス番号:")]

解 説

CALL COMSTOP 文を実行後、RS-232C にデータが来ても、CALL COMON が実行されるまで割り込みは保留されます。

デバイス番号

設定の対象となるカートリッジを 0(*)～9 の整数値で指定します。省略するときはカッコごと省略します。

文 例

CALL COMSTOP

もしくは

CALL COMSTOP("0:")

デバイス番号 0 を COMSTOP します。

CALL COMTERM

機 能

ターミナルモードにします。

書 式

CALL COMTERM[("[デバイス番号:]")]

解 説

コンピュータをターミナルモードにします。ターミナルモードとは、MSX をホストコンピュータの入出力端末として使用するモードです。ターミナルモードでは、MSX はホストコンピュータから受信したデータをディスプレイに表示します。そして、キーボードから入力されたデータを、ホストコンピュータに送信します。

1. ターミナルモードの起動

1. 接続する機器の仕様を確認する
2. CALL COMINI 命令による初期設定を行う
3. CALL COMTERM 命令で内蔵ターミナルソフトを起動する

RS-232C ポートを OPEN している場合には、CLOSE してから CALL COMTERM 命令を実行して下さい。

2. ターミナルモードの使用方法

ターミナルモードでは、以下のキーを使って、オプション機能を設定できます。

表 7.12　ターミナルモードのオプション機能一覧

キー	機能
[SHIFT] + [F1]	リテラルモードの切り替え
[SHIFT] + [F2]	エコーバックの切り替え
[SHIFT] + [F3]	プリンタエコーバックの切り替え
[STOP]	ブレーク信号の送信

1. リテラルモード

[SHIFT] + [F1] で ON、OFF を切り換えます。

[SHIFT] + [F1] を押すと、リテラルモードが ON になります。ON の時には、コントロールコード（&H1F 以下のコード）は、ハットマーク（^）とコントロールコードに 40H を足したコードの 2 文字で画面に表示されます。例えば&H01 なら

^A

と表示されます。

XON・OFF 制御を指定しているときは、XON と XOFF コードは表示されません。

もう一度、[SHIFT] + [F1] を押すと、リテラルモードは OFF になります。

2. ローカルエコーバック

[SHIFT] + [F2] で ON、OFF を切り換えます。

[SHIFT] + [F2] を押すと、エコーバックが ON になります。

ON の時には、キー入力した文字を送信すると同時に画面にも表示（エコーバック）します。

もう一度、[SHIFT] + [F2] を押すと、エコーバックは OFF になります。

3. プリンタエコーバック

[SHIFT] + [F3] で ON / OFF を切り換えます。

[SHIFT] + [F3] を押すと、プリンタへのエコーバックが ON になります。ON の時には、画面に出力したデータをプリンタにも出力します。

もう一度、[SHIFT] + [F3]を押すと、プリンタへのエコーバックは OFF になります。

4. ブレーク信号の送信

[STOP] キーを押してる間、ブレーク信号が送信されます。ブレーク信号とは、データ信号線（SD）が OFF になる状態です。

5. ターミナルモードの終了

[CTRL] + [STOP] キーを押すとターミナルモードは終了します。

ターミナルモードでは、

[F6]	リテラルモードの ON / OFF
[F7]	エコーバックの ON / OFF
[F8]	プリンタエコーの ON / OFF

となり、その機能の略号が画面に表示されます。そのほかのファンクションキーは、画面に表示されているファンクションキーの内容をそのまま送信します。

注　意　MSX では、画面表示の際に、DEL コードと BS (バックスペース) コードは同じ扱いをします。ホストコンピュータと接続した場合、ホストコンピュータによっては、行の先頭に DEL コードを送信するものがあり、画面表示が乱れることがあります。その場合、ホストコンピュータが DEL コードを送信しないように設定する必要があります。DEL コードの送信を止めることができないホストコンピュータと交信する場合は、ターミナルモードは使用できません。データの交信を行うプログラムを作成して、そこで DEL コードの処理をしてください。

3. ターミナルモードの例

```
call comini
Ok
call comterm
```

この例は、RS-232C を以下のように設定して、内蔵のターミナルソフトを起動する例です。

デバイス番号	0
キャラクタビット	8 ビット
パリティ	N
ストップビット長	1 ビット

XON/OFF 制御	X
RS/CS 制御	H
受信 LF 挿入	N
送信 LF 削除	N
SI/SO 制御	N
受信速度	1200bps
送信速度	1200bps
送信タイムアウト	0

設定の意味は、COMINI の項をご参照下さい。

4. エスケープシーケンス

RS-232C の内蔵ターミナルソフトがサポートするエスケープシーケンスは以下の通りです。

カーソル移動

<ESC>A　カーソルを上に移動
<ESC>B　カーソルを下に移動
<ESC>C　カーソルを右に移動
<ESC>D　カーソルを左に移動
<ESC>H　カーソルをホームポジションに移動
<ESC>Y<Y 座標+20H><X 座標+20H>　カーソルを(X,Y)の位置に移動

編集・削除

<ESC>j　画面をクリア
<ESC>l　行全体を削除
<ESC>E　画面をクリア
<ESC>K　行の終りまで削除
<ESC>J　画面の終りまで削除
<ESC>L　1行挿入
<ESC>M　1行削除

その他

<ESC>x4　カーソルの形を「■」にする
<ESC>x5　カーソルを消す
<ESC>y4　カーソルの形を「_」にする
<ESC>y5　カーソルを表示する

2. コマンド

LOAD

機　能

プログラムを受信します。

書　式

LOAD "COM[デバイス番号]：”[,R]

解　説

RS-232C から ASCII 形式の BASIC プログラムをロードします。EOF (&H1A) コードを受信するとロードを終了します。ロードする前のプログラムは消え、R オプションを指定していなかった場合、開いているファイルは閉じられます。通信誤りが検出された場合、

Device I/O error

のエラーメッセージが表示されます。

デバイス番号

設定の対象となるカートリッジを 0 (*) ～9 の整数値で指定します。

R オプション

ロード終了後に、プログラムが実行されます。開いているファイルは閉じません。

文　例

LOAD "COM:"

デバイス番号 0 から ASCII 形式の BASIC プログラムをロードします。

MERGE

機　能

プログラムを受信してメモリ上のプログラムとマージ混合します。

書　式

MERGE "COM[デバイス番号]："

解　説

RS-232C から ASCII 形式の BASIC プログラムをロードします。このときロードする前のプログラムは消去されずに、ロードされたプログラムと合併されます。EOF(&H1A) コードを受信すると、ロードは終了します。ロードする前のプログラムと同じ行番号がある場合は、ロードされたプログラムの行が残ります。通信誤りが検出された場合、

Device I/O error

のエラーメッセージが表示されます。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。

文　例

MERGE "COM:"

デバイス番号0から ASCII 形式の BASIC プログラムをロードしてマージします。

RUN

機　能

プログラムを受信した後に実行を開始します。

書　式

RUN "COM[デバイス番号]："[,R]

解　説

RS-232C から ASCII 形式の BASIC プログラムをロードして実行します。EOF（&H1A）コードを受信するとロードを終了します。ロードする前のプログラムは消去され、R オプションを指定していなかった場合、開いているファイルは閉じられます。通信誤りが検出された場合、

Device I/O error

のエラーメッセージが表示されます。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。

R オプション

開いているファイルは閉じません。

文　例

RUN "COM:"

デバイス番号0からASCII形式のBASICプログラムをロードして実行します。

SAVE

機　能

プログラムを送信します。

書　式

SAVE "COM[デバイス番号]："

解　説

RS-232CにASCII形式のBASICプログラムを送出します。通信誤りが検出された場合、

Device I/O error

のエラーメッセージが表示されます。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。

文　例

SAVE "COM:"

デバイス番号0からASCII形式のBASICプログラムを送出します。

3. ステートメント

CLOSE

機　能

RS-232C用ファイルをクローズします。

書　式

CLOSE [[#]ファイル番号[,[#]ファイル番号]...]

解　説

OPEN文でオープンされたファイル番号をクローズします。クローズしたファイル番号は、次にオープンするとき使用できます。ファイルが送信モードでオープンされている場合には、クローズするとEOFコードが送信されます。RUN、END、CLEAR、NEWなどのコマンドを実行した場合も、クローズされます。

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。省略すると全てのファイル番号がクローズされます。

文　例

CLOSE #1,#2

ファイル番号1と2をクローズします。

CLOSE

全てのファイル番号をクローズします。

INPUT

機　能

数値や文字を受信して変数に代入します。

書　式

INPUT #ファイル番号,変数[,変数]...

解　説

RS-232C からデータを受信して、変数に代入します。

ファイル番号

1 から MAXFILES 文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN 文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

変数

数値型変数、文字型変数、数値型配列変数、文字型配列変数を使用できます。
LINE INPUT # とは以下の点が異なります。

1. データの区切りとしては、カンマ（,）、空白（ただし、数値変数の場合）、CR コードが使用できます。
2. 数値変数が使用できます。

文　例

```
10 OPEN "COM:" FOR INPUT AS # 1
20 IF EOF (1) THEN GOTO 50
30 INPUT # 1,A$:PRINT A$
40 GOTO 20
50 CLOSE # 1
60 END
```

ファイル番号 1 からデータを取得して文字型変数 A$に代入して表示します。

LINE INPUT

機　能

254 文字までの文字を受信して文字型変数に代入します。

書　式

LINE INPUT #ファイル番号,変数

解　説

RS-232C から CR コードまでの文字列データを取得して、文字型変数に代入します。
INPUT #文と異なり、データの区切りは CR コードのみ有効です。

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

変数

文字型変数、文字型配列変数を使用できます。

文 例

```
10 OPEN "COM:" FOR INPUT AS #1
20 IF EOF(1) THEN GOTO 60
30 LINE INPUT #1,A$
40 PRINT A$
50 GOTO 20
60 CLOSE #1
70 END
```

ファイル番号1からデータを受信して文字列型変数A$に代入して表示します。

OPEN

機 能

RS-232C用ファイルをオープンします。

書 式

OPEN "COM[デバイス番号]:" [FOR モード] AS [#]ファイル番号

解 説

RS-232Cでデータの送受信をする前に、RS-232C用のファイルをオープンして、ファイル番号を割り当てます。

デバイス番号

設定の対象となるカートリッジを0(*)~9の整数値で指定します。

モード

OPEN時のモードを指定します。

OUTPUT　　送信モード

出力用に使用します。CLOSE するときに、EOF(&H1A)コードが送信されます。

INPUT　　受信モード

入力用に使用します。EOF (&H1A) コードを受信すると、それ以降のデータを受信することはできなくなりますから、必要な場合は、一度クローズして、オープンしなおして下さい。EOF コードを受信したかどうかは、EOF 関数で判ります。EOF コードは取得できません。

省略したとき　　送受信モード

CLOSE しても EOF コードは送信されません。また、EOF (1AH) コードを受信しても、そのまま渡されます。EOF 関数では必ず 0 を返します。

ファイル番号

1 から MAXFILES 文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。すでにオープンしているファイル番号を使用することはできません。

文　例

OPEN "COM:" FOR OUTPUT AS #1

デバイス番号 0 を送信モードでファイル番号 1 としてオープンします。

PRINT

機　能

数値や文字を送信します。

書　式

PRINT #ファイル番号,[式[セパレータ　式]...]

解　説

式で指定したデータを RS-232C へ送信します。

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で送信用もしくは送受信用にオープンしたファイル番号を指定します。

式

文字型、数値型の定数、変数、配列変数、式などが使用できます。

■数値
最初に1個の空白（数値が正のとき）かマイナス符号が送信され、次に数字が文字列に変換されて送信され、最後に1個の空白が出力されます。

■文字列
文字列が文字数だけ出力されます。

セパレータ

カンマ(,)とセミコロン(;)を使用できます。省略した場合には、CRコードとLFコードが出力されます。

■カンマ（,）
次のタブ位置までスペースを入れる。

■セミコロン（;）
何も出力せず、直後に次のデータを出力する。

文　例

```
10 OPEN "COM:" FOR OUTPUT AS #1
20 A$="ABC":B$="DEF"
30 PRINT #1,A$,B$
40 PRINT #1,A$;B$
50 PRINT #1,+50,-50
60 CLOSE #1
70 END
```

ファイル番号1に文字型変数の内容および数値型定数を送信します。実行結果は次のようになります。

```
ABC             DEF
ABCDEF
 50             -50
```

PRINT # USING

機 能

書式付き PRINT #

書 式

PRINT #ファイル番号, USING 書式記号； 式[,式]...

解 説

式で指定したデータを指定した書式で、RS-232C へ送信します。

ファイル番号

1から MAXFILES 文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN 文で送信用もしくは送受信用にオープンしたファイル番号を指定します。

書式記号

■文字列を編集するもの

!	文字列の左側1文字を表示する。
& 空白 &	文字列の左側から、指定した空白の数+2文字を表示する。
@	@を編集用として指定した文字列で置き換える。
#	数値を#で指定した桁数だけ表示する。
.	小数点の位置を指定する。
+ −	数値に正負符号をつける。書式の左端につけると数値の前に、右端につけると数値の後ろに正負符号をつける(ただし、−は左端にはつけられない)。
**	書式の左端を**にしておくと、数値の整数部の桁数が書式の指定より小さいときに、小さい分だけ*が表示される。
¥¥	書式の左端を¥¥にしておくと、数値の直前に¥がつけられる。¥¥は指数形式の書式指定をしているときには使えない。
**¥	書式の左端を**¥にしておくと、数値の直前に¥がつけられ、整数部の桁数が書式の指定より小さいときに、小さい分だけ*が表示される。**¥は指数形式の書式指定をしているときには使えない。
,	,を整数部の#と#の間、あるいは小数点の左側に置いたときには、整数部が3桁ごとにカンマで区切られて表示される。,は指数形式の書式指定をしているときには使えない。

^^^^　　^^^^を#の後ろにつけると、数値が指数形式で表示される。書式に＋または-を指定していない場合、数値が正ならば数値の前に空白が1つ、負なら数値の前に－が表示される。

%　　表示しようとする編集済みの数値が書式で指定した桁数より大きいと、数値の前の%がつけられる。

書式の中にこれらの記号以外の文字を置くと、文字の位置に応じて数値の前や後ろにその文字が表示されます。

式

文字型、数値型の定数、変数、配列変数、式などが使用できます。

文　例

```
10 A$="MSX-RS232C"
20 OPEN "COM:" FOR OUTPUT AS #1
30 PRINT #1,USING "!";A$
40 L = LEN(A$):IF L = 1 THEN GOTO 70
50 A$= RIGHT$(A$,L-1)
60 GOTO 30
70 CLOSE #1
80 END
```

ファイル番号1に文字型変数 A$の文字列から左側1文字ずつを送信します。

```
M
S
X
-
R
S
2
3
2
C
```

4. 関数

EOF

機能

EOF コード (&H1A) が受信されたかどうかを求めます。

書式

EOF(ファイル番号)

解説

EOF (&H1A) コードが受信され、それ以前のデータ全てをINTPUT$などにより読み込まれた場合は−1を、それ以外なら0を返します。送受信モードでオープンした場合には必ず0が返ります。

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

文例

```
IF EOF(1) THEN CLOSE #1
```

ファイル番号1の最後のデータが読みとられたら、そのファイルを閉じます。

INPUT$

機能

指定した数の文字を受信します。

書式

INPUT$(文字数,[#]ファイル番号)

解説

RS-232Cから指定した数の文字を受信します。

文字数

1から255までの整数値で指定できます。整数型定数、変数、配列変数、式などを使用できます。

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

文 例

```
10 OPEN "COM:" FOR INPUT AS #1
20 IF LOC(1)>=50 THEN X$=INPUT$(50,#1) ELSE GOTO 20
30 PRINT X$
40 GOTO 20
```

ファイル番号1から50文字を文字型変数X$に代入して表示します。

LOC

機 能

受信バッファの未取得文字数を求めます。

書 式

LOC(ファイル番号)

解 説

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

文 例

```
10 OPEN "COM:" FOR INPUT AS #1
20 IF LOC(1)>=1 THEN X$=INPUT$(1,#1) ELSE GOTO 50
30 PRINT X$
40 GOTO 20
50 IF EOF(1) THEN CLOSE #1 ELSE GOTO 20
60 END
```

ファイル番号1から未取得文字数を求めて1文字を文字列型変数X$に代入して表示します。

LOF

機　能

受信バッファの残りバイト数を求めます。

書　式

LOF(ファイル番号)

解　説

ファイル番号

1からMAXFILES文で指定された数までの整数値を指定できます。整数型定数、変数、配列変数、式などを使用できます。OPEN文で受信用もしくは送受信用にオープンしたファイル番号を指定します。

文　例

```
10 X0$= CHR$(&H11):X1$= CHR$(&H13)
20 CALL COMINI("0:8N1NNNN",300,,0)
30 OPEN "COM:" AS #1
40 IF LOC(1)= 1 THEN X$= INPUT$(1,# 1) ELSE GOTO 40
50 PRINT X$
60 IF LOF(1)<= 32 THEN PRINT # 1,X1$;
70 IF LOF(1)>= 120 THEN PRINT # 1,X0$;
80 GOTO 40
```

ファイル番号1の受信バッファの残りバイト数を求めて、32バイト以下になったらXOFFを送信し、120バイト以上になったらXONを送信します。

1.3 拡張BIOS

1.3.1 概要

MSX RS-232Cでは、アプリケーションソフトウェア用のサービスルーチンとして、拡張BIOSコールが用意されています。拡張BIOSコールにより、アプリケーションソフトウェアはそのスロットアドレスやアドレスなどの位置を調べ、インタースロットコール等により、ジャンプテーブルを経由して呼び出します。

高速処理を必要とする場合は、あらかじめスロットをイネーブルしておき、直接コールすることもできます。この章では、MSX RS-232C拡張BIOSを使用するのに必要な、拡張BIOSコールの方法と各BIOSの機能について解説します。

1.3.2 拡張BIOSの呼び出し

1. ジャンプテーブルアドレスの取得

アプリケーションは、まず以下の拡張BIOSコールにより、MSX RS-232C拡張BIOSの存在するスロットとジャンプテーブルの先頭アドレスを調べなければなりません。

拡張BIOSの存在するスロットとジャンプテーブルの先頭アドレスは、以下のようにして求めます。

1. RETURN情報エリア用のワークエリア（64バイト）をとる
2. 以下の設定を行い、FFCAH番地をコールする

コール手順

D	デバイス番号（8） MSX RS-232C拡張BIOSのデバイス番号は8
E	ファンクション番号（0）
B	RETURN情報エリアのスロットアドレス スロットアドレスは、システムワークエリアに保存されている
HL	RETURN情報エリアの先頭アドレス

RAMのスロットアドレスは以下のワークエリアに保存されています。このワークエリアはディスクが接続されているシステムで有効です。

表7.13 RAMのスロットアドレス

ページ	ワークエリアのアドレス
0	F341H
1	F342H
2	F343H
3	F344H

戻り値

B　次のRETURN情報エリアのスロットアドレス
HL　次のRETURN情報エリアの先頭アドレス

変更レジスタ

F

RETURN情報はアプリケーションが指定した領域に次のように格納されます。

B:HL→	ジャンプテーブルのスロットアドレス
	ジャンプテーブルの先頭アドレス(下位)
	ジャンプテーブルの先頭アドレス(上位)
	リザーブ(常に0)
B:HL returned→	

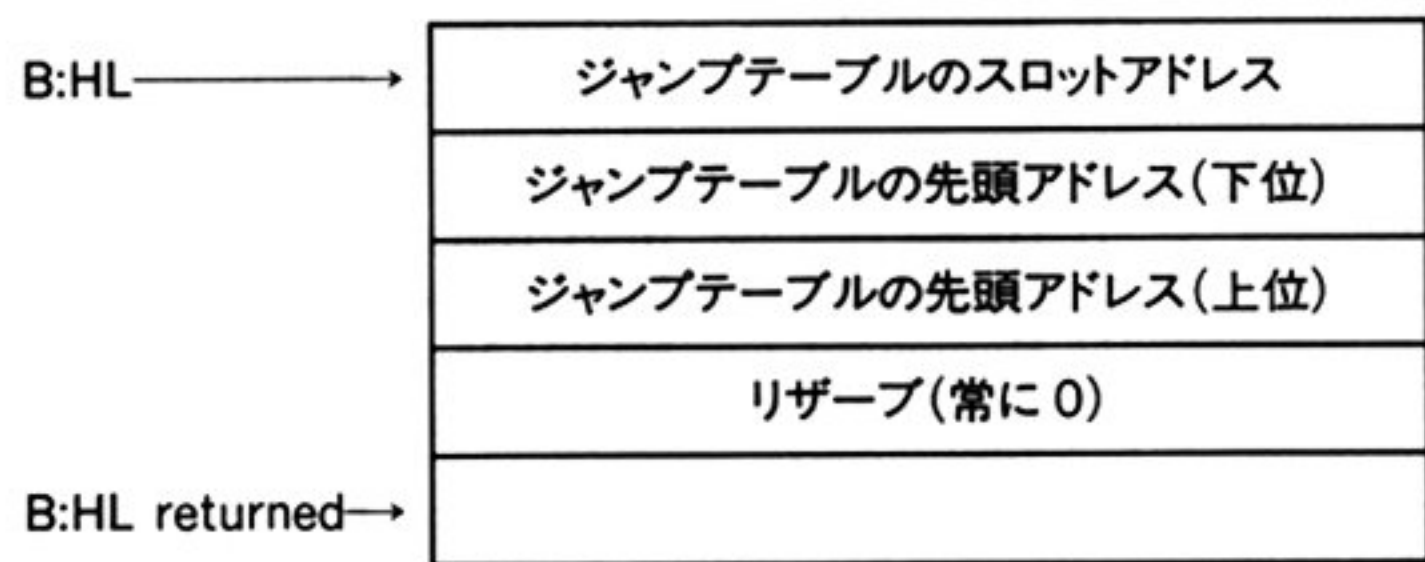



図7.3 RETURN情報の形式

RS-232Cが無いときは、BレジスタとHLレジスタの内容が変わらずに返ってきます。
スロットアドレスの表現はMSX共通で以下の通りです。

7	6	5	4	3	2	1	0
F	0	0	0	S	S	P	P

PP：基本スロット番号
SS：拡張スロット番号
F：拡張スロットを指定する場合に1

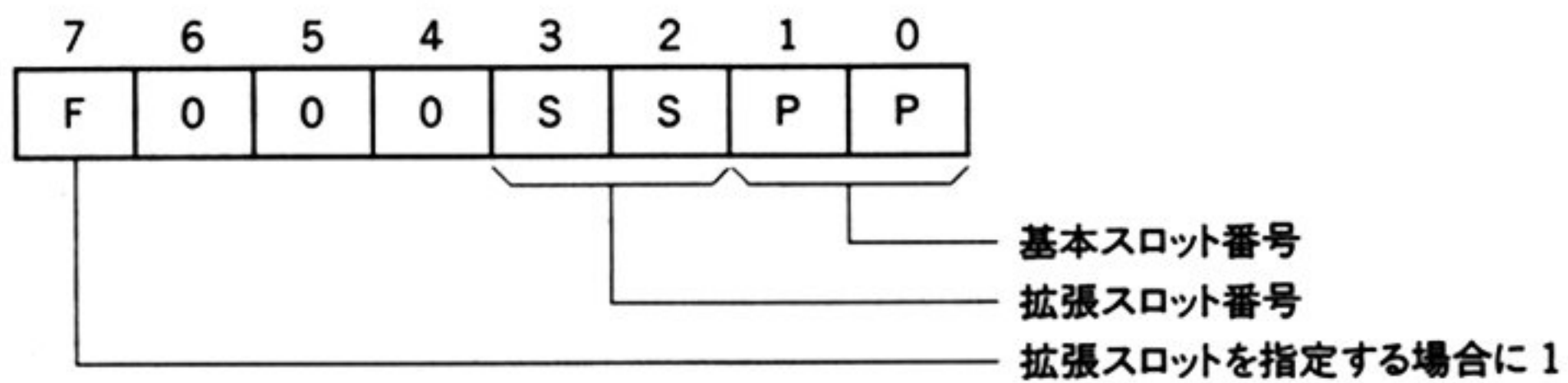



図7.4 スロットアドレスの形式

拡張 BIOS を使用する場合は、この拡張 BIOS コールで得られたジャンプテーブルをインタースロットコールなどにより呼び出し、目的の BIOS を使用します。

2. BIOS ジャンプテーブル

MSX RS-232C 拡張 BIOS は以下に示すジャンプテーブルを持っています。アプリケーションソフトウェアはインタースロットコール等で各エントリを呼び出す事により、BIOS の各機能を利用できます。

```
EXBTBL:  DEFB   DVINFB     ; Device information
         DEFB   DVTYPE     ; Device type
         DEFB   0          ; Reserved
         JP     INIT       ; Initialize RS-232C port
         JP     OPEN       ; Open RS-232C port
         JP     STAT       ; Read status
         JP     GETCHR     ; Receive data
         JP     SNDCHR     ; Send data
         JP     CLOSE      ; Close RS-232C port
         JP     EOF        ; EOF code received
         JP     LOC        ; Reports the number of characters
                           ; in the receiver buffer
         JP     LOF        ; Reports the number of free spaces
                           ; left in the receiver buffer
         JP     BACKUP     ; Back up a character
         JP     SNDBRK     ; Send break character
         JP     DTR        ; Turn DTR(ER) line on/off
         JP     SETCHN     ; Set channel number
                           ; (multi type only)
         RET               ; Reserved for MODEM cartridge
         RET               ; Reserved for MODEM cartridge
         RET               ; Reserved for MODEM cartridge
         RET               ; Reserved for MODEM cartridge
         RET               ; Reserved for MODEM cartridge
         RET               ; Reserved for MODEM cartridge
```

DVINFB

DVINFB (Device INFormation Byte) は以下の構成でそのカートリッジの仕様を表わします。

7	6	5	4	3	2	1	0
0	0						0

- ビット0: システム予約
- ビット1: 送信割り込み
- ビット2: BREAK キャラクタ検出
- ビット3: タイマインタラプト
- ビット4: キャリアデテクト(CD)
- ビット5: リングインジケータ(RI)
- ビット6: システム予約
- ビット7: システム予約

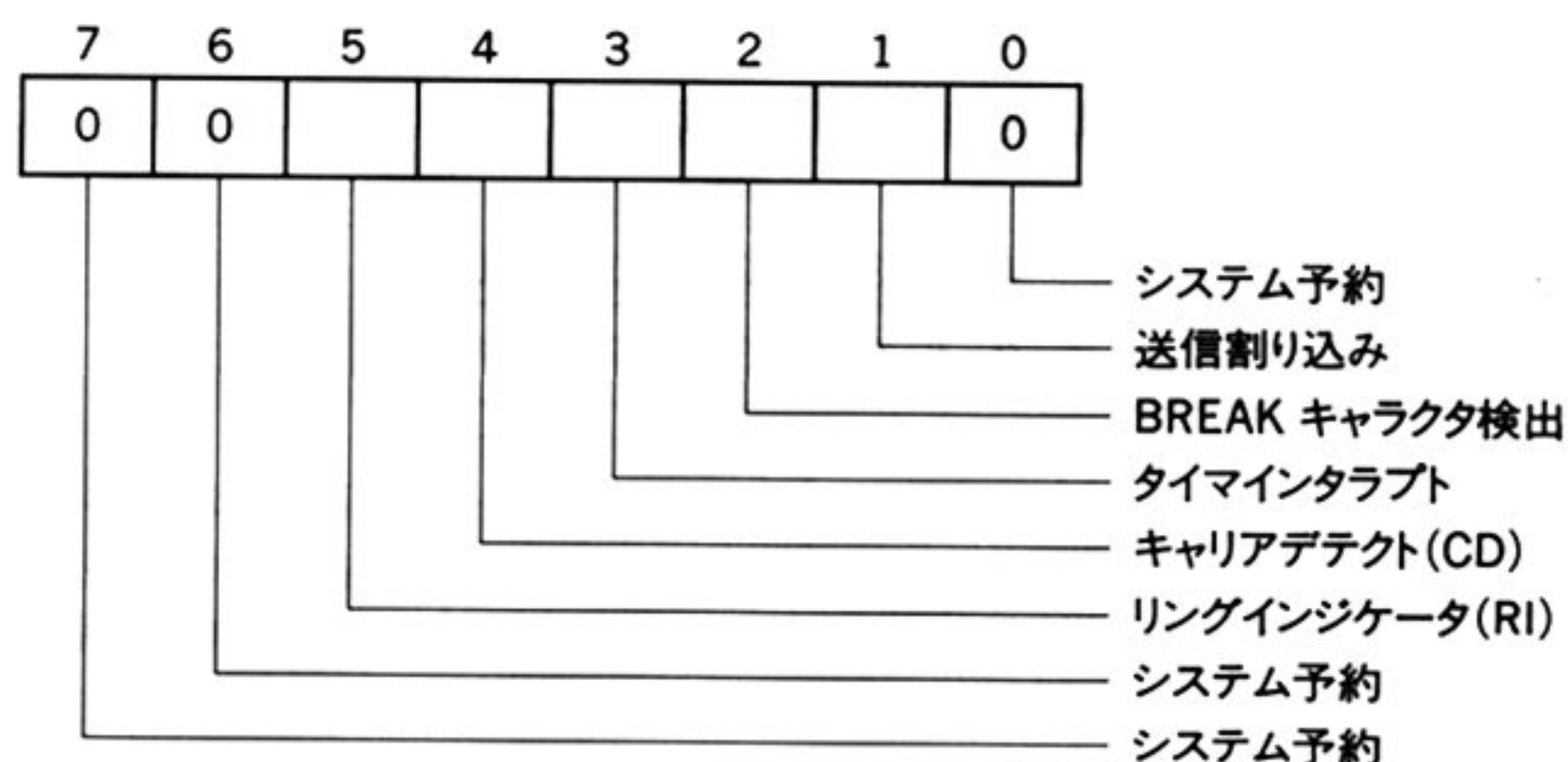



図7.5 DVINFB の形式

各ビットは1でその機能があることを示し、0でないことを示します。

DVTYPE

DVTYPEが0であればシングルチャンネルタイプのRS-232Cカートリッジです。この場合はSETCHNのエントリはありません。

DVTYPEが0以外であればマルチチャンネルタイプのRS-232Cカートリッジです。

3. BIOSの各機能

MSX RS-232C拡張BIOSの各機能はインタースロットコールにより呼び出されます。インタースロットコール（CALSLT）の呼び出しアドレスは001CHです。

コール手順

IY	上位8ビットにスロットアドレス
IX	呼び出しアドレス

その他のレジスタは機能により異なります。各項を参照してください。

INIT

機　能

RS-232C 拡張 BIOS をイニシャライズします。

コール手順

B　　パラメータテーブルのスロットアドレス
HL　　パラメータテーブルのアドレス

B:HL→

オフセット	意味	
+0	キャラクタ長	5～8
+1	パリティ	E、O、I、N
+2	ストップビット	1、2、3
+3	XON/XOFF 制御	X、N
+4	RS/CS 制御	H、N
+5	受信 LF 挿入	A、N
+6	送信 LF 削除	A、N
+7	SI/SO 制御	S、N
+8	受信速度	(16ビット)
+10	送信速度	(16ビット)
+12	送信タイムアウト	0～255

図 7.6　INIT パラメータテーブルの形式

オフセット＋0 から＋7 は ASCII キャラクタ、＋8 から＋12 まではバイナリです。ASCII キャラクタはすべて大文字とします。

戻り値

CY フラグ　パラメータの指定に間違いがある場合、1 にセットされます。

変更レジスタ

AF

解　説

イニシャライズデータの意味を説明します。

キャラクタ長

通信するデータのビット長を指定します。

5	5ビット
6	6ビット
7	7ビット
8	8ビット

パリティ

データに付加するパリティの指定をします。

N	パリティなし
E	偶数パリティ
O	奇数パリティ
I	送信時にはパリティビットは0 受信時には無視する

ストップビット

ストップビット長を指定します。

1	ストップビット1ビット
2	ストップビット1.5ビット
3	ストップビット2ビット

XON / XOFF 制御

XON / XOFF 制御を行なうかどうかの指定です。

X	XON / XOFF 制御を行なう
N	行なわない

XON / XOFF 制御とは、受信バッファがいっぱいになりそうなときに、XOFF (11H) コードを通信相手に送って送信を中止させ、バッファが空くとXON (13H) コードを送って送信を再開させる方式です。また、XOFF コードを受け取るとそれ以降の送信は中断され、XON コードを受け取ると送信が再開されます。

RS / CS 制御　H、N

RS/CS 制御を行なうかどうかの指定です。Hが指定されると、キャラクタを送信するときに、CSがチェックされます。

H	RS/CS 制御を行なう
N	行なわない

RS/CS 制御とは、受信バッファがいっぱいになりそうなときにRSをOFFにして送信を中止させ、バッファが空くとRSをONにして送信を再開させる方式です。また、CS

が OFF であると送信は中断され、CS が ON になると送信が再開されます。

受信 LF 挿入

CR（0DH）コードを受信したときにその後に LF（0AH）が受信されたものとみなす制御を行なうかどうかの指定です。

A	LF を挿入する
N	挿入しない

送信 LF 削除

CR（0DH）コードを送信した直後の LF（0AH）コードは送信せずに捨てる制御を行なうかどうかの指定です。

A	LF を削除する
N	削除しない

SI / SO 制御

データ長が 7 ビットのときに 0A0H コードから 0FFH のコードを送受信するとき文字セットの切り替えの制御を行なうかどうかの指定です。SI（0FH）コードにより G0 集合を呼び出し、SO（0EH）コードにより G1 集合を呼び出します。

S	SI / SO 制御を行なう
N	行なわない

受信 / 送信速度

通信速度をビット / 秒で以下の中から指定します。

50、75、110、300、600、1200、1800、2000、2400、3600、4800、7200、9600、19200

なお、受信速度、送信速度はそれぞれ別々に設定できます。

送信タイムアウト

データを送信するときに、XOFF コードが受信されていたり CS が OFF だったりしてデータを送信できない場合の待ち時間を指定します。単位は秒で範囲は 0 から 255 までです。0 を指定した場合はタイムアウトせずに送出できるまで待ち続けます。

注 意

このエントリは他の機能を呼び出す前に、必ず一度は呼び出されなくてはなりません。

OPEN

機　能

RS-232C をオープンします。

コール手順

HL　　FCB のアドレス（8000H 番地以上のアドレスを指定する）

C　　バッファ長（範囲は 32 文字〜127 文字）

E　　オープンモード

1　　<INPUT>モード

2　　<OUTPUT>モード

4　　<RAW> and <INPUT/OUTPUT>モード

FCB の取り方

HL⟶

9bytes for File Control
[C]×2 Bytes Receiver Buffer

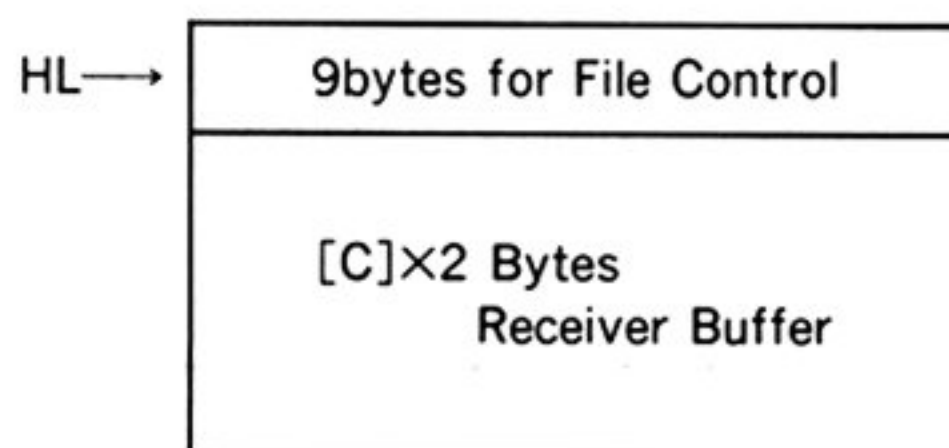



図 7.7　FCB の形式

戻り値

CY フラグ　オープンモードが正しくない場合は 1 にセットされます。

変更レジスタ

AF

解　説

この機能は RS-232C のすべての入出力処理に先立って行う必要があります。FCB エリアは 9 バイトのワークエリアと受信するデータの 2 倍の長さのバッファが必要です。受信データは受信データそのものと、エラー情報の 2 バイト構成でバッファに格納されます。このバッファ長は C レジスタで指定されます。FCB エリアは 8000H 番地以上に置かなければいけません。これは RS-232C カートリッジが割り込みによって呼び出されたときに、スロット間リード／ライトによるオーバーヘッドを避けるために FCB エリアを直接アクセスしているためです。

入出力モード

■ Input モード

このモードでは EOF (1AH) コードが受信され、それが GETCHR で渡されるときにキャリーフラグが1になり、それ以降の入力は CLOSE して OPEN し直さない限り EOF しか返されません。

■ Output モード

このモードでは CLOSE が呼ばれたときに自動的に EOF (1AH) コードを送出します。

■ Raw and input / output モード

このモードでは EOF (1AH) コードに関係なく入出力が行えます。

注　意

XON / XOFF 制御を行なう場合は入出力モードに関係なく行なわれます。

ポートが OPEN されるとハードウェアの受信割り込みが許可されます。一度 OPEN された後、なんらかのデータが受信バッファに残っている状態で CLOSE されると、その後再び OPEN しても残っていたデータは失われます。

STAT

機　能

ハードウェアの状態を返します。

コール手順

なし

戻り値

HL　　ステータス（信号は ON で、状態は検出されると 1)

bit 0	CD 信号
bit 1	RI 信号
bit 2	ブレーク信号検出 (*)
bit 3	DR 信号
bit 4	システム予約
bit 5	システム予約
bit 6	システム予約
bit 7	CS 信号
bit 8	システム予約
bit 9	システム予約

bit 10　[CTRL]+[STOP]キー検出 (*)
bit 11　パリティエラー (*)
bit 12　オーバーランエラー (*)
bit 13　フレーミングエラー (*)
bit 14　タイムアウトエラー (*)
bit 15　バッファオーバーフロー (*)

bit 0からbit 7はLレジスタ、bit 8からbit 15はHレジスタ、サポートされていない機能のビットは0を返します。
(*) のビットは読み出されるとリセットされます。

変更レジスタ

なし

GETCHR

機　能

受信バッファから文字を読み出します。

コール手順

なし

戻り値

A	受信文字
Sフラグ	エラーがある場合1にセットされます。
CYフラグ	オープンモードが1 (Input mode) で、文字がEOFコードである場合1にセットされます。

変更レジスタ

なし

解　説

このBIOSを呼び出す時は、LOCを使用して受信バッファに文字があるかどうかを調べて、文字がある場合にのみ呼び出せます。文字がない状態で呼び出したときの結果は、保証しません。

SNDCHR

機　能

RS-232Cに文字を送出します。

コール手順

A　　送信文字

戻り値

Cフラグ　XOFFを受信していたりCSがONになっていなかったりしたときに、[CTRL]＋[STOP]が押されると1にセットされます。

Zフラグ　XOFFを受信していたりCSがONになっていなかったときに、INITで指定された時間が経過しても待機条件がクリアされなかった場合、タイムアウトして、1にセットされます。

変更レジスタ

なし

解　説

タイムアウトエラーが発生したとき、および[CTRL]＋[STOP]キーが押された場合、文字は送信しません。

CLOSE

機　能

RS-232Cポートをクローズします。

コール手順

なし

戻り値

OPEN時にOutputモードに設定されていた場合

CYフラグ　XOFFを受信していたりCSがONになっていなかったりしたときに、[CTRL]＋[STOP]が押されると1にセットされます。

Z フラグ　XOFF を受信していたり CS が ON になっていなかったときに、INIT で指定された時間が経過しても待機条件がクリアされなかった場合、1 にセットされます。

OPEN 時に Input モードに設定されていた場合
なし

変更レジスタ

AF

解　説

FCB が解放されます。OPEN 時に OUTPUT モードの指定がされていた時は、EOF コード（1AH）が送出されます。
タイムアウトエラーが発生する可能性があるのは、この EOF コードを送出するときだけです。
CLOSE されると、ハードウェアの受信割り込みが禁止されます。

EOF

機　能

次の受信文字が EOF かどうかをチェックします。

コール手順

なし

戻り値

HL　EOF の場合、−1 でさらにキャリーフラグが 1 にセットされます。
EOF でない場合もしくはまだ受信バッファに文字がない場合、0 でさらにキャリーフラグが 0 にリセットされます。
RAW モードで RS-232C ポートがオープンされたときは、EOF は常に 0 で、キャリーフラグが 0 にリセットされます。

変更レジスタ

AF

LOC

機　能

受信バッファの中にある文字数を返します。

コール手順

なし

戻り値

HL　　受信バッファ内の文字数

変更レジスタ

AF

解　説

この機能が返す値にはバックアップされていた文字が含まれます。デバイスがInputモードでオープンされている場合にはEOF以降の文字は含まれません。ただし、この文字もバッファのスペースを必要とします。

LOF

機　能

受信バッファの空き容量を返します。

コール手順

なし

戻り値

HL　　受信バッファの空き容量

変更レジスタ

AF

解　説

この機能が返す値は、OPENの時に指定した受信バッファの文字数から上記LOCの値を引いた値に1を加えた（バックアップ文字の分）値です。

BACKUP

機　能

文字を受信バッファに戻します。これは1文字のみ可能です。

コール手順

C　　バックアップする文字

戻り値

なし

変更レジスタ

AF

SNDBRK

機　能

指定した数のブレーク文字を送出します。

コール手順

DE　　送出するブレーク文字数

戻り値

Cフラグ　[CTRL]＋[STOP]が押されると1にセットされます。

変更レジスタ

AF、DE

解　説

ブレーク信号送出中に[CTRL]＋[STOP]が押されると、キャリーフラグをセットしてアボートリターンします。

DTR

機　能

DTR（ER）をON/OFFします。

コール手順

A　　　コントロール指定
0　　　DTR（ER）をOFFします。
0以外　DTR（ER）をONします。

戻り値

なし

変更レジスタ

F

SETCHN

機　能

マルチチャンネルタイプのRS-232Cカートリッジで、制御の対象とするチャンネルを指定します。

コール手順

E　　　チャンネル番号

戻り値

Cフラグ　チャンネル番号が正しくない場合に1が返されます。

変更レジスタ

AF、BC

解　説

チャンネル番号はリセット時に0に設定されます。

注 意

ここでのチャンネル番号はBASICでのCOM0:などの番号とは異なり、マルチチャンネル RS-232C カートリッジのポートだけに付けられた番号です。つまりシングルチャンネルタイプの RS-232C カートリッジがマルチチャンネルタイプのものより小さいスロット番号にあっても、チャンネル番号は0から始まります。

1.4 サンプルプログラム

添付のフロッピーディスクの中に、RS-232C 拡張 BIOS を使ったサンプルプログラムが入っています。ファイル名は、「TERMINAL.MAC」です。

MSX-DOS からこのプログラムを起動すると、簡単なターミナルモードになります。終了するには CTRL + STOP を押してください。

このプログラムを作成するには MSX-DOS TOOLS を使って、次の操作を行います。

```
A>m80 = b:sample
No Fatal error(s)
A>l80 b:sample,b:sample/n/e
MSX-L.80   1.00 01-Apr-85   (c) 1981,1985 Microsoft
DATA      0103     0267      <   356>
XXXXX Bytes Free
[0000    0267            2]
A>
```

2章 MSX MODEM

2.1 ハードウェア

2.1.1 基本構成

MSX MODEM の基本ハードウェア構成は下図のとおりです。シリアルインターフェイスとして i8251 相当、ROM 32K バイト、RAM (バッテリバックアップ) 8K バイト、NCU 制御のためのパラレルポートとして i8255 相当を基本前提としています。ただし、後述の BIOS により、この基本ハードウェアとは異なった構成のハードウェアにも移植が可能です。

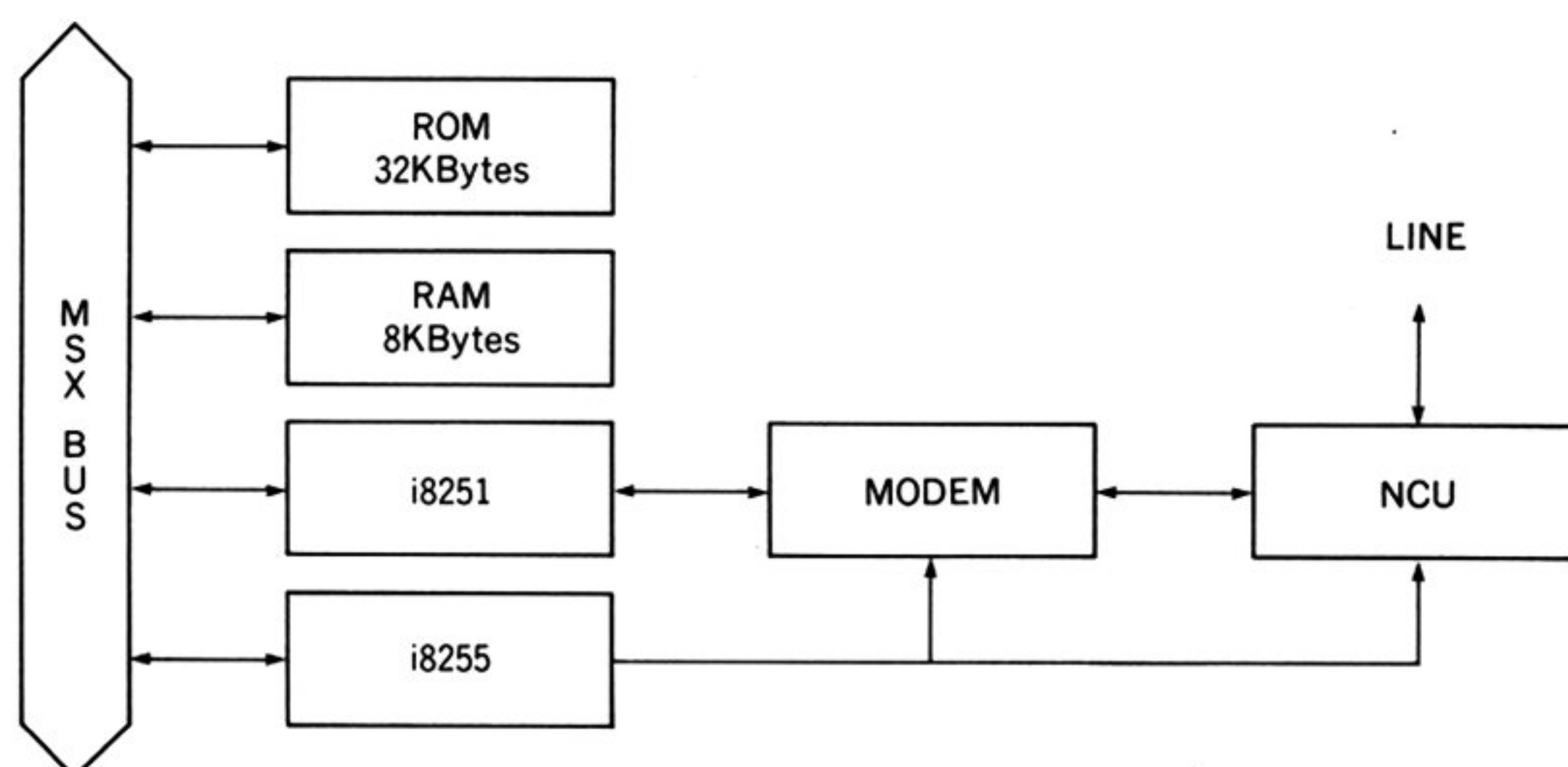



図 7.8　MSX MODEM の基本構成

短縮ダイヤルやシステムの設定値を記憶するためのバッテリバックアップされた CMOS

RAMはオプション設定とします。ソフトウェアではファイルとしてのデータアクセスをサポートするため容量は任意ですが、ディレクトリ領域を必要とすることなどから少なくとも2Kバイト程度が実用限界と思われます。単にシステムのコンフィギュレーションを記憶しておくメモリスイッチとしてなら必要最小限の容量でも動作可能です。なお、短縮ダイヤル用メモリは1件につき約100バイト程度の容量が必要です。

2.2 NCUの構成

NCUは図8.9のハードウェアを想定しています。この図は想定しているすべてのハードウェア機能を記載していますので、必ずしもこのような構成が実際的であるわけではありません。

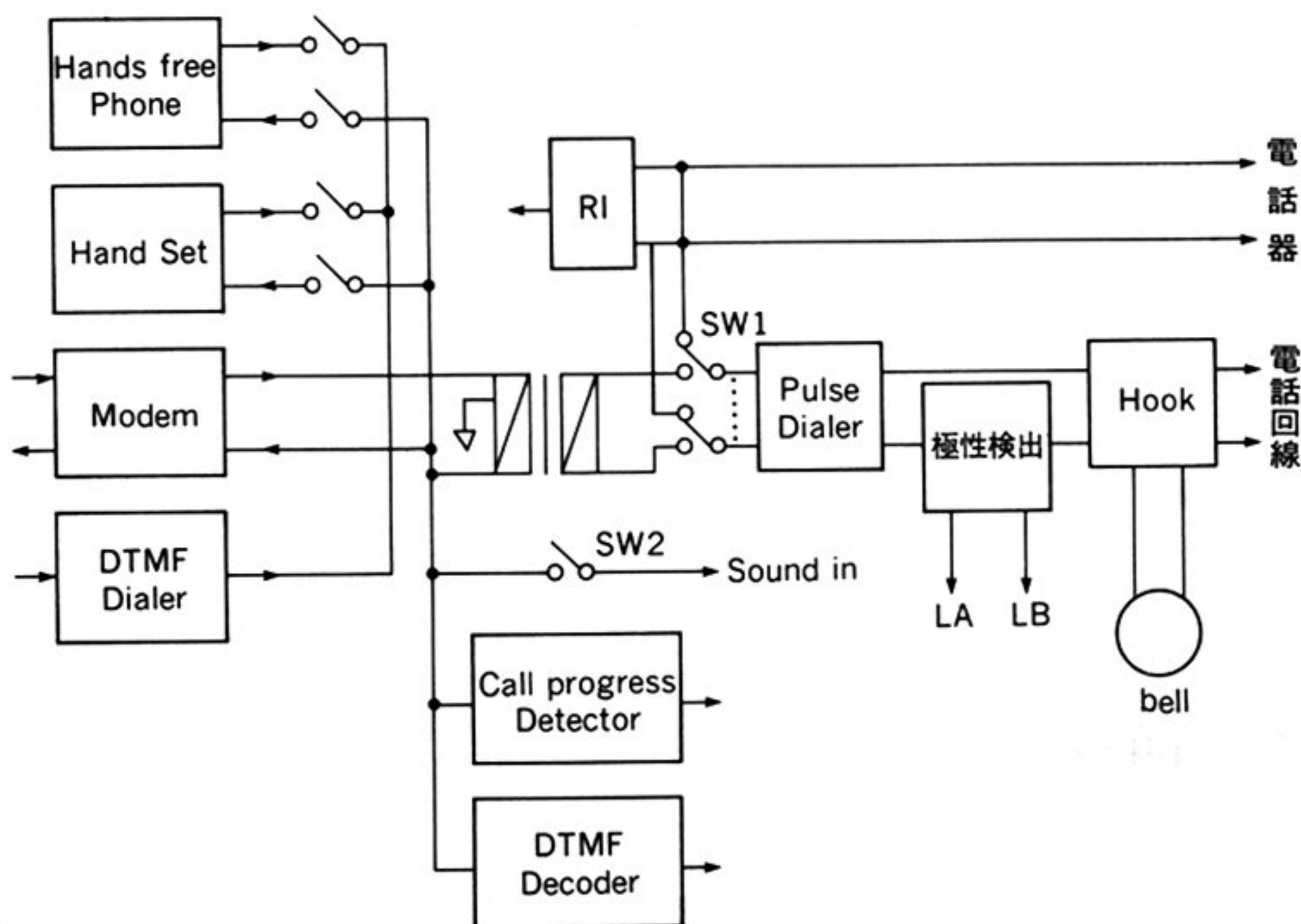



図7.9　NCUの構成

SW1は回線をモデムダイアラに接続するか、外部電話機に接続するかを選択します。SW2は電話線上の音を装置付属のスピーカまたはMSXのカートリッジバスを通じてTVのスピーカでモニタするかどうかを選択します。RI信号はリングパルスをワンショットなどにより波形整形されたものとします。したがって、リング信号が存在するあいだ1が返されます。内蔵ハンドセット、内蔵ハンズフリーフォンは他の電話機能、たとえば音楽を回線に送り出したりするものと置き換えることができます。

2.3 メモリの構成

MSX MODEM のシステムソフトウェアが実行するためには、図8.10で示すメモリ構成が必要です。ここで言うBANKとはMSX本来のBANKではなく、同一スロット内でローカルに拡張されたBANKを意味します。

8000H
RAM
6000H
ROM BANK0
ROM BANK1
ROM BANK2
ROM BANK3
4000H
2000H
0000H

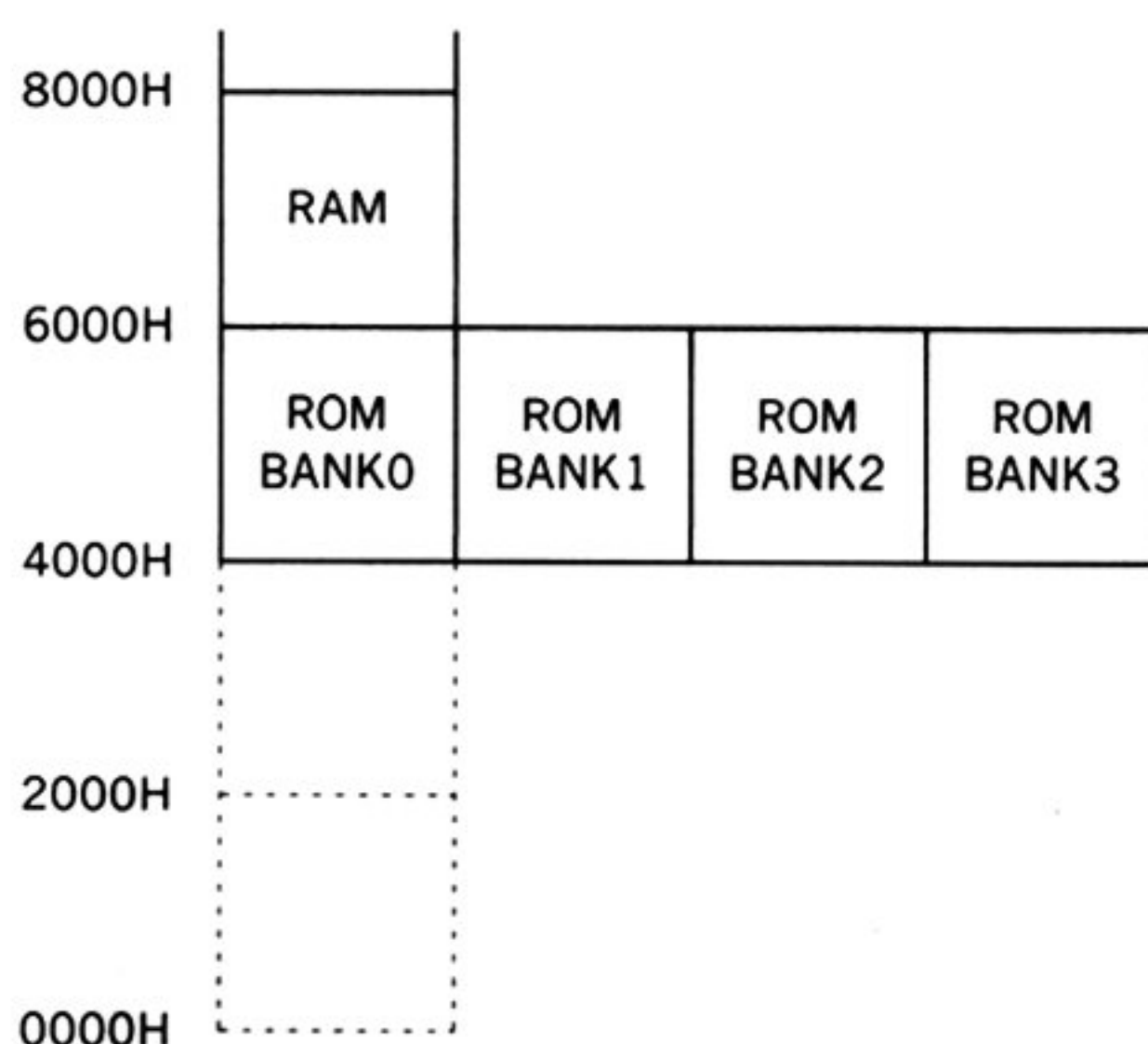



図7.10 メモリの構成

2.4 I/Oの構成

I/Oレジスタはすべてメインメモリ空間上に配置しなければなりません。アドレスの40H番地からFFH番地はMSXシステム標準装置のための領域ですので、その装置が標準装置として認められない限り使うことができません。

00番地から3F番地は規定されていませんので使用することが可能ですが、規定されていないために、他のものと同じ番地を割り当ててしまいますと、システムが動作しない事態がおこってしまいます。

これを避けるために、必ずメモリ空間に配置して下さい。

2.5 拡張 BASIC

2.5.1 概要

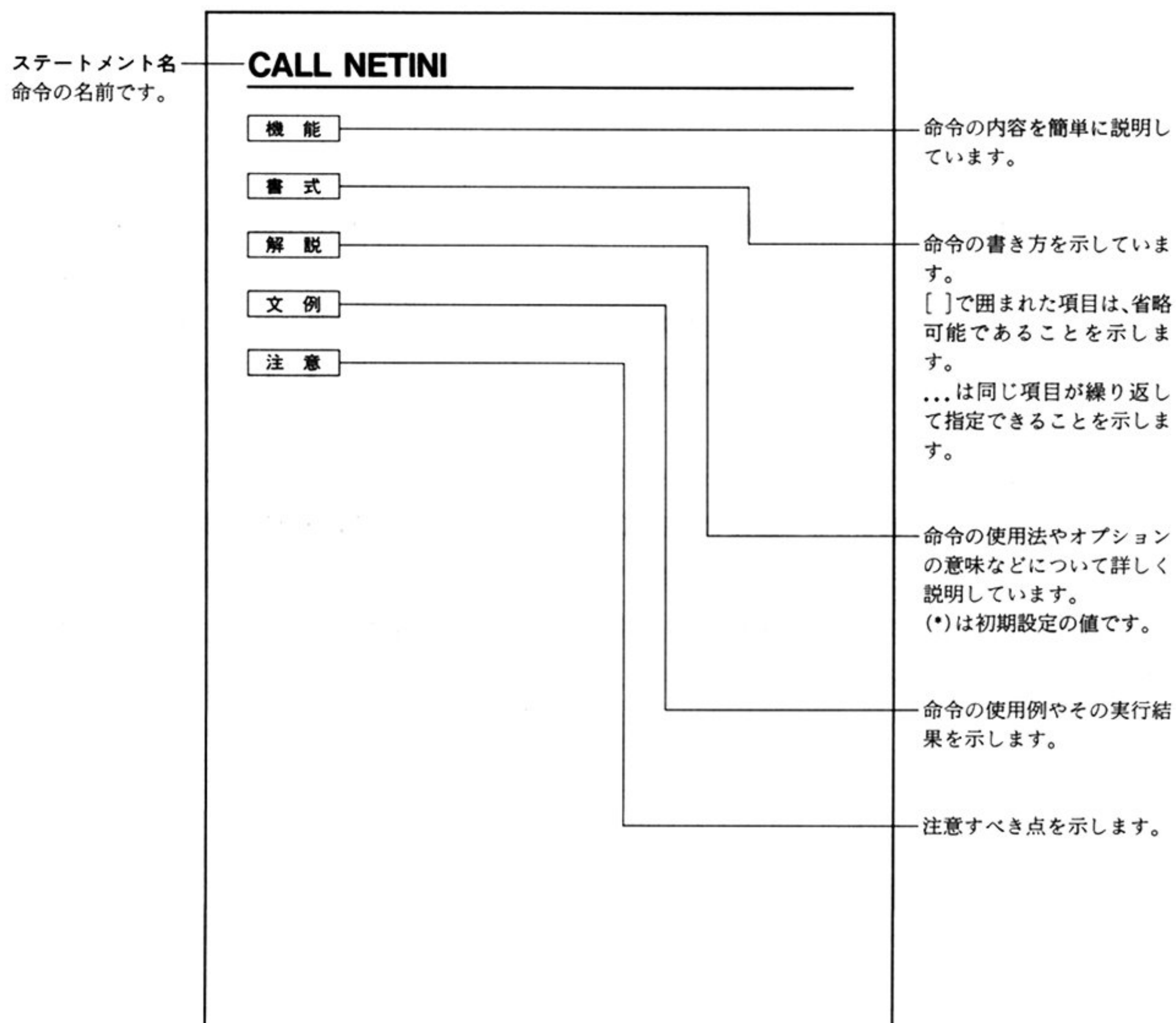
MSX MODEM には、各機能を簡単に使用できるように、MSX MODEM 拡張 BASIC が用意されています。コマンド、ステートメント、関数は MSX BASIC、MSX Disk BASIC と同じですが、書式や指定方法の違いにより、通信が行なえるように機能が拡張されています。

使い方は、CALL COMINI のように拡張ステートメントの形式です。CALL は_（アンダーバー）で代用できます。

デバイス番号は、スロット番号の小さいスロットにさしこまれている通信用カートリッジの順に、0 から割り当てられます。MSX RS-232C カートリッジなど、MSX MODEM 以外の通信用カートリッジといっしょに使用された場合にも、同様にスロット順に 0 から割り当てられます。

通信用カートリッジには、RS-232C インターフェイス、MODEM カートリッジ、「MSX-SERIAL232」などがあります。

2.5.2 この章の表記法

ステートメント名
命令の名前です。
CALL NETINI
機 能
書 式
解 説
文 例
注 意
命令の内容を簡単に説明しています。
命令の書き方を示しています。
[ ]で囲まれた項目は、省略可能であることを示します。
...は同じ項目が繰り返して指定できることを示します。
命令の使用法やオプションの意味などについて詳しく説明しています。
(*)は初期設定の値です。
命令の使用例やその実行結果を示します。
注意すべき点を示します。

2.5.3 拡張 BASIC コマンド一覧

1. 拡張ステートメント（CALL 文と共に使用します）

コマンド名	機 能	ページ
COMBREAK*	電話回線にブレーク信号を送出します	153
COM GOSUB*	通信用ポートから割り込みがかかると指定したサブルーチンを実行します	153
COMHELP	COMINI の HELP メッセージを表示します	154
COMINI*	通信回線の初期化を行います	155
COMOFF*	通信用ポートからの割り込みを禁止します	158
COMON*	通信用ポートからの割り込みを許可します	159
COMPROTOCOL*	ファイル転送に用いるプロトコルを決定します	159
COMSTAT*	通信用ポートのステータスを求めます	161
COMSTOP*	通信用ポートからの割り込みを保留します	163
COMTERM*	ターミナルモードに切り換えます	164
DIAL	電話番号送出を行います	169
DIALC	1桁だけの電話番号を送出します	171
DTMF	DTMF デコーダよりデータを読み取ります	172
LINESEL*	回線の切り換えを行います	173
NETCARRIER	半2重通信時のキャリアを制御します	174
NETCONFIG*	ハードウェアのコンフィグレーションを返します	175
NET GOSUB	電話回線に着呼があったとき、または DTMF データが来たとき指定したサブルーチンを実行します	177
NETHOOK	回線の切断・接続を行います	178
NETINI*	NCU の初期設定を行います	179
NETMODEM	モデムの送出電力およびイコライザの ON/OFF を指定します	180
NETOFF	電話回線に着呼があったとき、または DTMF データが来たときの割り込みを禁止します	181
NETON	電話回線に着呼があったとき、または DTMF データが来たときの割り込みを許可します	182
NETSPK	スピーカを ON/OFF します	183
NETSTAT*	NCU のハードウェアの状態を返します	183
NETSTOP	電話回線に着呼があったとき、または DTMF データが来たときの割り込みを保留します	185

2. コマンド

コマンド名	機　能	ページ
LOAD*	BASIC プログラムを通信ポートからメモリにロードします	186
MERGE*	プログラムを通信用ポートからロードし、メモリ上のプログラムとマージ（混合）します	187
RUN*	BASIC プログラムを通信用ポートからメモリにロードし実行します	187
SAVE*	BASIC プログラムを通信用ポートに送ります	188

3. ステートメント

コマンド名	機　能	ページ
CLOSE*	OPEN 文で開いたファイルを閉じます	189
INPUT #*	OPEN 文で開かれたファイルからデータを順に読み取り、変数に代入します	190
LINE INPUT #*	254 文字までの文字列をファイルから読み取り、文字型変数に代入します	191
OPEN*	通信用ファイルを開きます	191
PRINT #*	OPEN 文で開かれたファイルにデータを書き込みます	192
PRINT # USING*	OPEN 文で開かれたファイルに指定した書式でデータを書き込みます	193

4. 関数

コマンド名	機　能	ページ
EOF*	ファイルの最後のデータが読み取られたかどうかを返します	196
INPUT$*	ファイルから指定した数の文字を入力します	196
LOC*	ファイルの中の文字数を返します	197
LOF*	ファイルの中の空きスペースの大きさを返します	198

*のステートメントは必須です。

その他は、ハードウェアがサポートしていなければ、「Illegal function call」になります。

デバイス番号が省略されたときは、0 となります。

また、COMSTAT と COMBREAK は、その実行前に「OPEN”COM :”」命令が実行されていなければ「File not open」になります。

ハードウェアがその機能をサポートしているか否かは、CALL NETCONFIG 命令で求められます。

2.5.4 拡張BASICの解説

1. 拡張ステートメント

CALL COMBREAK

機 能

電話回線にブレーク信号を送出します。

書 式

CALL COMBREAK[(["デバイス番号：”],式)]

解 説

式で指定された文字数文のブレーク信号を電話回線に出力します。ブレーク信号送出中、全ての送出データは強制的に0になります。
CALL COMBREAK文を実行する前にOPEN文でCOM用のファイルがオープンされていなければ「File not open」になります。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。省略するときはコロン（：）まで省略します。

式

数値定数、変数、配列変数、式で指定します。指定できる範囲は3から32767までの値です。省略した場合は、10になります。

文 例

CALL COMBREAK(''0:'',20)
20文字分のブレイク信号を電話回線に出力します。

CALL COM GOSUB

機 能

通信用ポートから割り込みがかかると指定したサブルーチンへ実行を移します。

書　式

CALL COM(["デバイス番号："],GOSUB 行番号)

解　説

CALL COMON 文を使って割り込みを許可してから、データが受信されると、割り込みがかかり、行番号で指定されたサブルーチンに実行が移ります。サブルーチンはRETURN 文で終了します。サブルーチンの実行終了後は割り込みが起こった時点で実行していたステートメントの次のステートメントに戻ります。
サブルーチン実行中は自動的に COMSTOP 状態になり、サブルーチン内で COMOFF しないかぎり、RETURN によって COMON 状態になります。

デバイス番号

設定の対象となるカートリッジを 0 (*) ～9 の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

行番号

0 から 65529 までの整数型定数で指定します。

文　例

CALL COM("0:",GOSUB 1000)
デバイス番号 0 から割り込みがかかると、行番号 1000 で始まるサブルーチンを実行します。

CALL COM(,GOSUB 2000)
デバイス番号 0 から割り込みがかかると、行番号 2000 で始まるサブルーチンを実行します。

CALL COMHELP

機　能

COMINI の HELP メッセージを表示します。

書　式

CALL COMHELP[("デバイス番号：")]

解　説

次の様にメッセージを表示します。

```
Initialize statement options
CALL COMINI("
<device # {0,1,2...9} > :
<character length {5,6,7,8} >
<parity {E,O,I,N} >
<stop bits {1,2,3} >
<XON / XOFF {X,N} >
<1 dummy {any character} >
<auto LF on receive {A,N} >
<auto LF on transmit {A,N} >
<SI / SO {S,N} >"
, <recieve speed(BPS) >
, <send speed(BPS) >
, <time out count> )
Default:
CALL COMINI("0:8NIX NNN",300 ,300 ,0)
```

デバイス番号

省略するとデバイス番号は0と見なされます。

文　例

```
CALL COMHELP("0:")
CALL COMHELP
```

デバイス番号0に差し込んである通信カートリッジのCOMONI文で設定できる通信モードの項目を表示します。

CALL COMINI

機　能

通信回線の初期化を行います。

書　式

CALL COMINI[(["[デバイス番号：][デバイス番号：][キャラクタ長[パリティ[ストップビット長[XON / XOFF 制御[ダミー[受信 LF[送信 LF[SI / SO 制御]]]]]]]]"][,[受信速度][,[送信速度][,[タイムアウト]]]])]

解　説

通信に必要なパラメータの設定を行います。COMINIを実行しなくても電源投入時、RESET時に省略値で初期設定されます。

デバイス番号

0（*）～9までの整数値を指定します。省略するときはコロン（：）まで省略します。省略するとデバイス番号は0:になります。

キャラクタ長

接続する機種に合わせて送信する1キャラクタのビット数を5、6、7、8の範囲で指定します。省略するとキャラクタ長は8になります。

5	5ビット
6	6ビット
7	7ビット
8（*）	8ビット

パリティ

パリティチェックの方法を指定します。

E	偶数パリティ
O	奇数パリティ
I	パリティを無視する。キャラクタ長が5、6、7ビットのときに有効。送信時にはパリティビットを0にする（キャラクタ長が8ビットの時は指定できない）
N（*）	パリティを使用しない

ストップビット長

ストップビットのビット数を指定します。

1（*）	1ビット
2	1.5ビット
3	2ビット

XON/OFF制御

XON/OFFによるフロー制御を行うかどうかを指定します。

X（*）	XON/OFF制御を行う
N	XON/OFF制御を行わない
	XONコード=&H11、XOFFコード=&H13

ダミー

RS-232C の拡張 BASIC と互換性を持たせるためのダミーデータです。スペースを設定して下さい。

受信 LF 挿入

CR (0DH) コードを受信したときに、CR コードと LF (0AH) コードに変換する制御です。

A	CR コードと LF コードに変換する
N (*)	CR コードと LF コードに変換しない

送信 LF 削除

CR (0DH) コードの次に LF (0AH) コードを続けて送信するとき、LF コードを削除して、CR コードのみを送信する制御です。

A	LF を削除する
N (*)	LF を削除しない

SI/SO 制御

キャラクタ長が 7 ビットのときに、080H から 0FFH コードに対して SI/SO 制御の指定を行います。

S	SI/SO 制御を行う
N (*)	SI/SO 制御を行わない

送信速度

送信側の速度を設定します。その速度がサポートされていない場合はシステムのデフォルト値または指定速度より遅いモデムの速度に読み換えられます。

受信速度

受信側の速度を設定します。その速度がサポートされていない場合はシステムのデフォルト値または指定速度より遅いモデムの速度に読み換えられます。

タイムアウト

XON/XOFF コントロールの待ち時間を 0 (*) ～255 の整数値で指定します。単位は約 1 秒です。省略した場合や、0 を指定した場合は、タイムアウトしません。

文 例

CALL COMINI("0:8N1X")

デバイス番号	0
キャラクタ長	8 ビット

パリティ	なし
ストップビット長	1ビット
XON／XOFF 制御	あり
受信 LF 挿入	なし
送信 LF 挿入	なし
SI／SO 制御	なし

CALL COMINI("7E3N AAS")

デバイス番号	0
キャラクタ長	7ビット
パリティ	偶数
ストップビット長	2ビット
XON／XOFF 制御	なし
受信 LF 挿入	あり
送信 LF 削除	あり
SI／SO 制御	あり

CALL COMOFF

機 能

通信用ポートからの割り込みを禁止します。

書 式

CALL COMOFF[("デバイス番号：")]

解 説

通信用ポートからの割り込みを禁止します。CALL COMOFF 後に通信ポートデータが来ても割り込みは発生しません。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文 例

CALL COMOFF("0:")
または
CALL COMOFF
デバイス番号0での割り込みを禁止します。

CALL COMON

機　能

通信用ポートからの割り込みを許可します。

書　式

CALL COMON[("デバイス番号：")]

解　説

CALL　COMON 実行後に通信用ポートにデータが来ると割り込みが発生し、CALL COM GOSUB 文で指定してある行のサブルーチンが実行されます。

デバイス番号

設定の対象となるカートリッジを0（*）〜9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文　例

CALL COMON("0:")

または

CALL COMON

デバイス番号0にデータが来ると、割り込みが発生します。

CALL COMPROTOCOL

機　能

ファイル転送に用いるプロトコルを決定します。

書　式

CALL　COMPROTOCOL[(["デバイス番号：][プロトコル]"][[,タイマ1][,タイマ2]])]

解　説

ファイル転送に用いるプロトコルを決定します。COMINI 文を実行しなくても電源投入時・RESET 時に省略値で初期設定されます。

デバイス記号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

プロトコル

T	テキスト転送（EOFを処理する）（*）
R	テキスト転送（たれ流し）
X	XMODEM（チェックサム）
C	XMODEM（CRCチェック）

タイマ1

プロトコルがTおよびRの場合は送出する文字間の時間となります。この値は1/60秒単位となります。省略時は0とします。
プロトコルがXおよびCの場合はXMODEMプロトコルのタイムアウト値となり、秒単位で指定します。省略時は10とします。
数値型変数、変数、配列変数、式で指定できます。

タイマ2

プロトコルがTおよびRの場合は送出する行間の時間となります。この値は1/60秒単位となります。省略時は0とします。
プロトコルがXおよびCの場合はXMODEMプロトコルのブロック監視タイマ値となり、秒単位で指定します。省略時は20とします。
数値型変数、変数、配列変数、式で指定できます。

COMTERMでファイルのUP/DOWN LOADを行う時、次のようにXON/XOFF制御の設定を行います。

T	XON/XOFF制御あり
R	XON/XOFF制御あり
X	XON/XOFF制御なし
C	XON/XOFF制御なし

文 例

CALL COMPROTOCOL("0:T")

デバイス番号0の通信カートリッジでファイルをテキスト転送します。

CALL COMSTAT

機 能

通信用ポートのステータスを求めます。

書 式

CALL COMSTAT(["デバイス番号："],数値変数)

解 説

通信用ポートのステータスを2バイトの数値型データとして返し、数値変数に代入します。

数値変数

整数型変数で、各ビットの意味は次のとおりです。

ビット0

キャリア信号の検出

0　　キャリアを検出していない

1　　キャリアを検出

ビット1

被呼表示（リングインジケータ）の検出

0　　リング信号がある間1になる

1　　サポートされていない場合は常に0を返す

ビット2

ブレーク信号の検出

0　　ブレーク信号を検出していない

1　　ブレーク信号を検出した

ビット3

データセットレディ

常に1

ビット4

NCU 機能

0	なし
1	あり

ビット5

未使用（常に0）

ビット6

未使用（常に0）

ビット7

未使用（常に0）

ビット8

未使用（常に0）

ビット9

Long loop 検出

0	Long loop は発生していない
1	Long loop は発生している

ビット10

[CTRL] + [STOP] キー検出

0	押されていなかった
1	押されていた

ビット11

パリティエラー

0	エラーなし
1	エラーあり

ビット12

オーバーランエラー

0	エラーなし
1	エラーあり

ビット 13

フレーミングエラー

0　　エラーなし

1　　エラーあり

ビット 14

タイムアウトエラー

0　　エラーなし

1　　エラーあり

ビット 15

オーバフローエラー

0　　エラーなし

1　　エラーあり

CALL COMSTAT 文を実行する前に、OPEN 文で COM 用のファイルがオープンされていなければ「File not open」になります。

デバイス番号

設定の対象となるカートリッジを 0（*）～9 の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

文　例

```
CALL COMSTAT("0:",F):PRINT BIN$(F)
```

または

```
CALL COMSTAT(,F):PRINT BIN$(F)
```

通信ポート 0 のステータスを 2 進法で変数 F に代入して画面に表示します。

画面には、例えば 10000011000（上位バイト　00000100、下位バイト 00011000）と表示されます。

CALL COMSTOP

機　能

通信用ポートからの割り込みを保留します。

書 式

CALL COMSTOP[("デバイス番号:")]

解 説

CALL COMSTOP 実行後に通信用ポートにデータが来ても次に CALL COMON が実行されるまで割り込みが保留されます。

デバイス番号

設定の対象となるカートリッジを0(*)~9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文 例

CALL COMSTOP("0:")
または
CALL COMSTOP
デバイス番号0からの割り込みを保留します。

CALL COMTERM

機 能

ターミナルモードに切り換えます。

書 式

CALL COMTERM[("[デバイス番号:][文字セット[インターレース[漢字コード[DEL 受信処理]]]]")]

解 説

コンピュータをターミナルモードに切り換えます。この時、回線はモデム側に切り換えます。また、HOOK は OFF に切り換えます。回線が接続されていない場合は、「Device I/O Error」となります。
ターミナルモードは CTRL + STOP で終了します。
ターミナルモードに入る前に、COM 用のファイルをオープンしている場合は、「File already open」になります。

デバイス番号

設定の対象となるカートリッジを0(*)~9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン(:)まで省略します。

文字セット

表示に使用する文字フォントを指定します。

A　ANK（*）。フォントサイズは6×8。

M　MSXキャラクタ（含ひらがな）。フォントサイズは8×8。

K　漢字。フォントサイズは全角が16×16、半角が8×16。漢字ROMが実装されている時のみ有効で、その他の場合は無視される。

G　グレイスケール漢字。フォントサイズは全角が12×16、半角が6×16。MSX2、MSX2+でグレイスケール漢字ROMが実装されている時のみ有効で、その他の場合は無視される。

S　12ドット漢字。フォントサイズは全角が12×12、半角が6×12。MSX2、MSX2+で12ドット漢字ROMが実装されている時のみ有効で、その他の場合は無視される。

インタレースモード

ノンインタレースかインタレースかを指定します。インタレースの指定はMSX2、MSX2+で文字セットが漢字、グレイスケール漢字、および12ドット数字の時有効で、その他の時は無視します。

N　ノンインタレース（*）

I　インタレース（MSX2で漢字のとき有効）

漢字コード

漢字コードを指定します。漢字コードの指定は文字セットが漢字、グレイスケール漢字、および12ドット漢字で、日本語入力フロントエンド（MSX-JE）がサポートされている時のみ有効です。それ以外の時は無視されます。

O　旧JISコード

J　新JISコード

S　シフトJISコード（*）

N　NEC漢字コード

DEL受信処理

B　バックスペース

N　NULL（*）

1. ターミナルモードの使用方法

ターミナルモードでは、以下のキーを使って、オプション機能を設定できます。

表7.14 ターミナルモードのオプション機能

キー	機能
SHIFT + F1	リテラルモードの切り替え
SHIFT + F2	エコーバックの切り替え
SHIFT + F3	プリンタエコーバックの切り替え
STOP	ブレーク信号の送信
SHIFT + F4	アップロード
SHIFT + F5	ダウンロード
SHIFT + SELECT	ターミナルモードの一時中断
GRAPH + SELECT、CTRL + SPACE	日本語入力フロントエンド実行/終了

1. リテラルモード

SHIFT + F1 でON、OFFを切り換えます。

ONの時は、アスキーコードの1FH以下に割り当てられているコントロールコードは、すべてハットマーク(^)とコントロールコードに40Hを足したコードで表される文字として画面に表示されます。例えば、CRコード(0DH)は、

^M

と画面に表示されます(Mのコードは4DH)。

XON・OFF制御を指定しているときは、XONとXOFFコードは表示されません。

もう一度、SHIFT + F1 を押すと、リテラルモードはOFFになります。

2. ローカルエコーバック

SHIFT + F2 でON、OFFを切り換えます。

ONの時は、キーボードから入力したデータは、相手のコンピューターに送られるととにもう一度、SHIFT + F2 を押すと、エコーバックはOFFになります。

画面に表示されます。

3. プリンタエコーバック

SHIFT + F3 でON/OFFを切り換えます。

ONの時は、画面に表示されたデータがプリンタに出力されます。10秒以上プリンタに出力されない状態が続くとこのモードは解除されます。

もう一度、SHIFT + F3 を押すと、プリンタへのエコーバックはOFFになります。

4. ブレーク信号の送出

STOP キーを押すと、ブレーク信号が100文字分出力され、送信データが0になります。

5. アップロード

SHIFT + F4 でON / OFFを切り換えます。

ONの時は、ファイルを相手のコンピュータに送ります。再度このキーを押すとアップロードを中断します。ただし、XMODEMを使用している場合は中断できません。

6. ダウンロード

SHIFT + F5 でON / OFFを切り換えます。

ONの時は、ファイルを相手のコンピュータから受け取ります。再度このキーを押すとダウンロードを中断します。ただし、XMODEMを使用している場合は中断できません。

7. ターミナルモードの一時中断

SHIFT + SELECT でON / OFFを切り換えます。

ONの時に、ターミナルモードを中断します。この時回線は保持されます。

8. 日本語入力フロントエンド実行/終了

GRAPH + SELECT または CTRL + SPACE でON / OFFを切り換えます。

ONの時、日本語漢字入力フロントエンドの実行を開始します。

2. 各モードの設定条件

1. MSX1

表7.15 MSX1時のスクリーンモード

指　定	COMTERM入力時のスクリーンモード			
	0	1	2	3
AN、AI	AN2	AN2	AN2	AN2
MN、MI	MN2	MN2	MN2	MN2
KN、KI	KN2	KN2	KN2	KN2
GN、GI	KN2	KN2	KN2	KN2
SN、SI	SN2	SN2	SN2	SN2

2. MSX2（VRAM64K）

表7.16 VRAM64K時のスクリーンモード

指　定	画面サイズ	COMTERM入力時のスクリーンモード						
		0	1	2	3	4	5	6
AN、AI	1～40	AN2	AN2	AN2	AN2	AN2	AN5	AN6
	41～80	AN6	－	AN2	AN2	AN2	AN5	AN6
MN、MI	1～40	MN2	MN2	MN2	MN2	MN2	MN5	MN6
	41～80	MN6	－	MN2	MN2	MN2	MN5	MN6
KN	1～40	KN2	KN2	KN2	KN2	KN2	KN5	KN6
	41～80	KN6	－	KN2	KN2	KN2	KN5	KN6
KI	1～40	KN2	KN2	KN2	KN2	KN2	KI5	KI6
	41～80	KN6	－	KN2	KN2	KN2	KI5	KI6
GN	1～40	KN2	KN2	KN2	KN2	KN2	GN6	GN6
	41～80	GN6	－	KN2	KN2	KN2	GN6	GN6
GI	1～40	KN2	KN2	KN2	KN2	KN2	GI6	GI6
	41～80	GI6	－	KN2	KN2	KN2	GI6	GI6
SN	1～40	KN2	KN2	KN2	KN2	KN2	SN5	SN6
	41～80	SN6	－	KN2	KN2	KN2	SN5	SN6
SI	1～40	KN2	KN2	KN2	KN2	KN2	SI5	SI6
	41～80	SI6	－	KN2	KN2	KN2	SI5	SI6

3. MSX2、MSX2＋(VRAM128K)

表 7.17　VRAM128K 時のスクリーンモード

指　定	画面サイズ	COMTERM 入力時のスクリーンモード								
		0	1	2	3	4	5	6	7	8
AN、AI	1～40	AN2	AN2	AN2	AN2	AN2	AN5	AN6	AN7	AN5
	41～80	AN7	−	AN2	AN2	AN2	AN5	AN6	AN7	AN5
MN、MI	1～40	MN2	MN2	MN2	MN2	MN2	MN5	MN6	MN7	MN5
	41～80	MN7	−	MN2	MN2	MN2	MN5	MN6	MN7	MN5
KN	1～40	KN2	KN2	KN2	KN2	KN2	KN5	KN6	KN7	KN5
	41～80	KN7	−	KN2	KN2	KN2	KN5	KN6	KN7	KN5
KI	1～40	KN2	KN2	KN2	KN2	KN2	KI5	KI6	KI7	KI5
	41～80	KI7	−	KN2	KN2	KN2	KI5	KI6	KI7	KI5
GN	1～40	KN2	KN2	KN2	KN2	KN2	GN6	GN6	GN6	GN6
	41～80	GN6	−	KN2	KN2	KN2	GN6	GN6	GN6	GN6
GI	1～40	KN2	KN2	KN2	KN2	KN2	GI6	GI6	GI6	GI6
	41～80	GI6	−	KN2	KN2	KN2	GI6	GI6	GI6	GI6
SN	1～40	KN2	KN2	KN2	KN2	KN2	SN5	SN6	SN7	SN5
	41～80	SN7	−	KN2	KN2	KN2	SN5	SN6	SN7	SN5
SI	1～40	KN2	KN2	KN2	KN2	KN2	SI5	SI6	SI7	SI5
	41～80	SI7	−	KN2	KN2	KN2	SI5	SI6	SI7	SI5

3. ターミナルモードの例

CALL COMTERM(''0:KI'')

CALL COMTERM(''KI'')

この例は、デバイス番号0の通信カートリッジで、キャラクタを漢字、インタレースでターミナルモードを実行します。

CALL DIAL

機　能

電話番号送出を行います。

書　式

CALL DIAL(["デバイス番号：”],電話番号[,モード])

解 説

電話番号の文字列にしたがって回線をダイヤルに接続し、約3秒後に（ただし、コールプログレストーン検出機能がある場合は、ダイヤルトーン検出後すみやかに）ダイヤル送出を開始します。
回線使用中および回線が接続されていない場合は「Device I/O　Error」となります。
ダイヤル終了後、回線はモデム側のままです。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

電話番号

電話番号として使用できるのは以下の半角文字です。

0から9までの整数	
H	約1秒間ON HOOKします。
A～D、#、*	DTMFのみ有効です。
<	3秒間ポーズを作ります。
：	第2ダイヤルトーンを待ちます。トーン検出機能のない場合は<と同じです。
T	トーンダイヤルへの切り換えを行います。

モード

電話回線の種類を数値定数、変数、配列変数、式で指定します。

0	プッシュボタン（DTMF）
1	ダイヤルパルス（10pps）
2	ダイヤルパルス（20pps）
3	自動検出
省略	初期設定値（モデムのハードウェア設定による）

サポートされていない機能を選択したときは「Illegal function call」になります。

文 例

```
CALL DIAL("0:","0:03-123-4567",0)
```

デバイス番号0のカートリッジから、03-123-4567に電話をかけます。

```
A$="03 123 4567":CALL DIAL(, A$)
```

デバイス番号0のカートリッジから、03-123-4567に電話をかけます。

注 意

ハードウェアがサポートされてないときは、「Illegal function call」になります。

CALL DIALC

機 能

1桁だけの電話番号を送出します。

書 式

CALL DIALC(["デバイス番号："],1桁の電話番号[,モード])

解 説

1桁だけダイヤル送出を行います。2桁以上は「Illegal function call」になります。回線をダイヤラに接続します。
回線が接続されていない場合は、「Device I/O Error」となります。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

電話番号

電話番号として使用できるのは以下の半角文字です。

0から9までの整数	
H	約1秒間ON HOOKします。
A～D、#、*	DTMFのみ有効です。
<	3秒間ポーズを作ります。
：	第2ダイヤルトーンを待ちます。トーン検出機能のない場合は<と同じです。

上記以外の文字は無視しますが、サポートされていない機能（文字）を指定したときは「Illegal function call」になります。ただし、a～d、hは大文字に読み換えます。

モード

電話回線の種類を数値定数、変数、配列変数、式で指定します。

0	プッシュボタン（DTMF）
1	ダイヤルパルス（10pps）

2　　　ダイヤルパルス（20pps）
省略　　初期設定値（モデムのハードウェア設定による）

サポートされていない機能を選択したときは「Illegal function call」になります。

文 例

CALL DIALC(''0:'',''3'',0)

デバイス番号0の通信カートリッジから番号"3"を電話回線に送り出します。使用する電話はプッシュホンとします。

注 意

ハードウェアでサポートされてないときは、「Illegal function call」になります。

CALL DTMF

機 能

DTMFデコーダよりデータを読み取ります。

書 式

CALL DTMF(["デバイス番号："],文字変数)

解 説

DTMFデコーダよりデータを読み取り、文字変数に代入します。データが来ていない場合は来るまで待ちます。
文字変数には最大32バイトのDTMFコードが入りますが、それ以上のDTMFコードは無視します。
[CTRL]＋[STOP]の入力があった場合は中止します。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

文 例

CALL DTMF(''0:'',D$):PRINT D$
または
CALL DTMF(, D$):PRINT D$

デバイス番号0の通信カートリッジからDTMF信号を読み込み、D$に代入します。その後に、その値を画面に表示します。

注 意

ハードウェアでサポートされてないときは、「Illegal function call」になります。

CALL LINESEL

機 能

回線の切り換えを行います。

書 式

CALL LINESEL([″デバイス番号：″],数値変数)

解 説

電話回線を電話機に接続するか、内蔵モデムに接続するかを選択します。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

数値変数

各ビットの意味は次のとおりです。

ビット0

0	内蔵モデムに接続
1	電話機に接続

ビット1

1	内蔵ハンドセットのマイクに接続

ビット2

1	内蔵ハンドセットのスピーカに接続

ビット3

1	内蔵ハンズフリーフォンのマイクに接続

ビット4

1	内蔵ハンズフリーフォンのスピーカに接続

サポートされていない機能のビットを1にしたときは「Illegal function call」になります。
数値変数を省略したときは「Syntax error」になります。

文 例

CALL LINESEL("0:",0)
電話回線をデバイス番号0の通信カートリッジ側に切り換えます。

A =&B11000:CALL LINESEL(, A)
電話回線を電話機側に切り換えます。

注 意

ハードウェアでサポートされてないときは、「Illegal function call」になります。

CALL NETCARRIER

機 能

半2重通信時のキャリアを制御します。

書 式

CALL NETCARRIER(["デバイス番号:"],数値変数[[,タイマ1][,タイマ2]])

解 説

半2重通信時（V23およびV27terモデムを利用している時）のキャリアON/OFFを制御します。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（:）まで省略します。

数値変数

数値型変数、変数、配列変数、式で指定できます。

0	キャリアをOFFする
1	キャリアをONする
省略	「Syntax error」になる

タイマ1

このコマンドがコールされてからモデムのRS信号をONにするまでの時間となります。この値は10m秒単位となります。

数値型変数、変数、配列変数、式で指定できます。

タイマ2

モデムのCS信号がONになってからNETCARRIERコマンドがリターンするまでの時間なります。この値は10m秒単位となります。

数値型変数、変数、配列変数、式で指定できます。

文　例

CALL NETCARRIER(,1,2,2)

デバイス番号の通信カートリッジのキャリアをONします。

この時ガードタイムT1およびT2を20mSとします。

CALL NETCONFIG

機　能

ハードウェアのコンフィグレーションを返します。

書　式

CALL NETCONFIG(["デバイス番号:"],モード，数値変数)

解　説

ハードウェアのコンフィグレーションを数値変数に代入して返します。数値変数は、サポートしているものについて対応するビットが1になります。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（:）まで省略します。

モード0

ビット0	Bell 103	300bps	Full duplex
ビット1	Bell 212A	1200bps	Full duplex
ビット2	CCITT V21	300bps	Full duplex
ビット3	CCITT V22	1200bps	Full duplex

ビット4　　CCITT V22bis　　2400bps Full duplex
ビット5　　CCITT V23　　1200bps Half duplex
ビット6　　CCITT V27ter　　4800bps Half duplex
ビット7　　CCITT V29　　9600bps Half duplex
ビット8　　CCITT V32　　9600bps Full duplex
ビット9〜ビット15は0に固定

モード1

ビット0　　プッシュボタン（DTMF）
ビット1　　ダイヤルパルス（10pps）
ビット2　　ダイヤルパルス（20pps）
ビット3　　自動検出
ビット4　　A〜Dをサポートしている。
ビット5　　Hをサポートしている。
ビット6　　DTMF-パルスの切り換えがソフトでできます。
ビット7　　10pps、20ppsの切り換えがソフトでできます。
ビット8〜ビット15は0に固定

モード2

ビット0　　外部電話機
ビット1　　内蔵モデム
ビット2　　内蔵ハンドセット
ビット3　　内蔵ハンズフリーフォン
ビット4〜ビット15は0に固定

モード3

ビット0　　リング信号検出
ビット1　　Call progress 検出
ビット2　　回線極性検出
ビット3　　課金パルス検出
ビット4　　DTMFデコーダ
ビット5　　スピーカ
ビット6　　ONFF HOOK 機能
ビット7　　外部電話機ONFF HOOK 検出
ビット8　　MSX標準カートリッジ
ビット9　　RS-232C
ビット10　　送出電力切替機能
ビット11　　キャリア制御作用

ビット12　　Long loop 検出機能
ビット13～ビット15は0に固定

モード0～3以外

数値変数=0（固定）

文　例

CALL NETCONFIG("0:",1,F):PRINT BIN$(F)

デバイス番号0の通信カートリッジにおいてモード1のステータスを変数Fに代入して、2進法で表示します。

返される値は、

上位バイト　　00000000
下位バイト　　11100111

CALL NETCONFIG(,3,F):PRINT BIN$(F)

デバイス番号0の通信カートリッジにおいてモード3のステータスを変数Fに代入して、2進法で表示します。

返される値は、

上位バイト　　00000010
下位バイト　　00100001

CALL NET GOSUB

機　能

電話回線に着呼があったとき、またはDTMFデータが来たとき指定したサブルーチンへ実行を移します。

書　式

CALL NET(["デバイス番号:"],GOSUB 行番号)

解　説

CALL NETON 文を使って割り込みを許可してから、電話回線に着呼があるかDTMFデータが受信されると、割り込みがかかり、行番号で指定されたサブルーチンに実行が移ります。サブルーチンはRETURN文で終了します。サブルーチンの実行終了後は、割り込みがかかった時点で実行していた次のステートメントに戻ります。

着呼による割り込みか、DTMFデータによる割り込みかの区別は、CALL NETSTAT

文を実行しリターン値のビット8とビット0で行います。
サブルーチン実行中は自動的にNETSTOP文により割り込み保留となります。サブルーチン中でNETOFF文を指定しない限り、RETURN文により再び割り込み許可状態になります。

デバイス番号
設定の対象となるカートリッジを0 (*) ～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

行番号
0から65529までの整数型定数で指定します。

文 例

CALL NET("0:",GOSUB 1000)
デバイス番号0の通信カートリッジに着信かDTMFの入力があった場合は、行番号1000に実行を移します。

注 意

リングインジケータとDTMFデコーダが両方ともサポートされていないときは「Illegal function call」になります。

CALL NETHOOK

機 能

回線の切断・接続を行います。

書 式

CALL NETHOOK(["デバイス番号："],数値変数)

解 説

デバイス番号
設定の対象となるカートリッジを0 (*) ～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

数値変数

0　　ON HOOK（受話器を置き、回線が切れた状態）

1　　OFF HOOK（受話器をあげ、回線が接続された状態）
省略　「Syntax error」になる

それ以外の値を指定したときは、「Illegal function call」になります。

文　例

CALL NETHOOK("0:",0)
または
CALL NETHOOK(,0)

デバイス番号0の通信カートリッジのフックをONにします。

注　意

ハードウェアでサポートされてないときは、「Illegal function call」になります。

CALL NETINI

機　能

NCUの初期設定を行います。

書　式

CALL NETINI[(["[デバイス番号：][通信モード]"][,ダイアラモード][,モデム])]

解　説

NCUの初期設定を行います。NETINIを実行しなくても電源投入時、RESET時に省略値で初期設定されます。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

通信モード

A　　アンサモード
O　　オリジネートモード（*）

ダイヤラモード

0　　プッシュボタン（DTMF）
1　　ダイヤルパルス（10pps）

2	ダイヤルパルス（20pps）
3	自動検出
省略	初期設定値（モデムのハードウェア設定による）

サポートされていない機能を選択したときは「Illegal function call」になります。

モデム

0	Bell　103	300bps　Full duplex
1	Bell　212A	1200bps Full duplex
2	CCITT V21	300bps　Full duplex
3	CCITT V22	1200bps Full duplex
4	CCITT V22bis	2400bps Full duplex
5	CCITT V23	1200bps Half duplex
6	CCITT V27ter	4800bps Half duplex
7	CCITT V29	9600bps Half duplex
8	CCITT V32	9600bps Full duplex
省略	初期設定値	

サポートされていないモデムを選択したときは「Illegal function call」になります。

文　例

CALL NETINI("A",0)

デバイス番号0の通信カートリッジをアンサーモード、使用している電話回線をプッシュホンに設定します。

CALL NETMODEM

機　能

モデムの送出電力およびイコライザのON/OFFを指定します。

書　式

CALL NETMODEM[(["デバイス番号:"][,送出電力][,イコライザ])]

解　説

モデムキャリアの出力電力およびイコライザ制御を指定します。キャリアの出力値は回線インピーダンス600Ω時の電力とします。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

送出電力

送出電力×（－1）の値を設定します。15がデフォルトで－15dBmが指定されたことになります。

数値型定数、変数、配列変数、式で指定できます。

イコライザ

数値型定数、変数、配列変数、式で指定できます。

0　　イコライザを使用しない（デフォルト）

1　　イコライザを使用する

2　　自動的にイコライザを制御する

例

```
CALL NETMODEM(,5)
```

送出電力を、－5 dBm にします。

CALL NETOFF

機 能

電話回線に着呼があったとき、またはDTMFデータが来たときの割り込みを禁止します。

書 式

CALL NETOFF[("デバイス番号：")]

解 説

CALL NETOFF後電話回線に着呼があるか、またはDTMFデータが来ても割り込みは発生しません。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文 例

```
CALL NETOFF("0:")
CALL NETOFF
```

注 意

リングインジケータとDTMFデコーダが両方ともサポートされていないときは「Illegal function call」になります。

CALL NETON

機 能

電話回線に着呼があったとき、またはDTMFデータが来たときの割り込みを許可します。

書 式

CALL NETON[("デバイス番号:")]

解 説

CALL NETON実行後電話回線に着呼があるか、またはDTMFデータが来ると割り込みが発生し、CALL NET GOSUB文で指定してある行のサブルーチンが実行されます。

デバイス番号

設定の対象となるカートリッジを0(*)〜9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文 例

```
CALL NETON("0:")
CALL NETON
```

注 意

リングインジケータとDTMFデコーダが両方ともサポートされていないときは「Illegal function call」になります。

CALL NETSPK

機　能

スピーカをON/OFFします。

書　式

CALL NETSPK(["デバイス番号："],数値変数)

解　説

カートリッジスロットのSUNDINと電話回線を接続・切断します。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン（：）まで省略します。

数値変数

0	スピーカOFF
1	スピーカON
省略	「Syntax error」になる

それ以外の値を指定したいときは、「Illegal function call」になります。

文　例

```
CALL NETSPK("0:",0)
A = 0:CALL NETSPK(,A)
```

デバイス番号0の通信カートリッジからの音を切ります。

注　意

ハードウェアがサポートされていないときは「Illegal function call」になります。

CALL NETSTAT

機　能

NCUのハードウェアの状態を返します。

書 式

CALL NETSTAT(["デバイス番号:"],数値変数)

解 説

NCUのステータスを求め、変数に代入します。

デバイス番号

設定の対象となるカートリッジを0(*)～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはコロン(:)まで省略します。

数値変数

整数型変数で、各ビットの意味は次のとおりです。

ビット15～ビット9

未使用(常に0)

ビット8

DTMFデータの受信

0	DTMFデータは来ていない
1	DTMFデータが来ている

ビット7

外部電話機のHOOK状態検出

0	ON HOOK
1	OFF HOOK

ビット6

Call progress

1	400Hzのトーンを検出した

ビット5

課金パルス検出

極性反転をラッチする。リードリセット。モデムのハードウェアがサポートしているときに有効。

ビット4、ビット3

回線極性検出

0	0	loop off

0	1	DC loop (LB)
1	0	DC loop (LA)

ビット2、ビット1

ダイヤラモード

0	0	プッシュボタン (DTMF)
1	0	ダイヤルパルス (10pps)
0	1	ダイヤルパルス (20pps)
1	1	自動検出

ビット0

リングインジケータ

リング信号をラッチする。

0	サポートされていない
1	リング信号がある。リードリセット。

文 例

```
CALL NETSTAT("0:",F):PRINT BIN$(F)
CALL NETSTAT(,F):PRINT BIN$(F)
```

デバイス番号0の通信カートリッジのNCUステータスを変数Fに代入して、2進法で表示します。

CALL NETSTOP

機 能

電話回線に着呼があったとき、またはDTMFデータが来たときの割り込みを保留します。

書 式

CALL NETSTOP[("デバイス番号:")]

解 説

CALL NETSTOP実行後電話回線に着呼があるか、またはDTMFデータが来ても、次にCALL NETONが実行されるまで割り込みが保留されます。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。省略するときはカッコごと省略します。

文 例

```
CALL NETSTOP("0:")
CALL NETSTOP
```

注 意

リングインジケータとDTMFデコーダが両方ともサポートされていないときは「Illegal function call」になります。

2.コマンド

LOAD

機 能

BASICプログラムを通信ポートからメモリにロードします。

書 式

LOAD "COM[デバイス番号]:"[,R]

解 説

LOAD実行前のプログラムを消去し、開いているファイルを閉じてから、アスキー形式のBASICプログラムをロードします。EOFコード（1AH）を受信するとロードを終了します。Rオプションを指定するとロード終了後プログラムが実行されます。この場合、開いているファイルは閉じません。

プログラム転送に際しエラーコレクションは行いません。

文 例

```
LOAD"COM0:",R
```

MSX BASICのプログラムを通信用ポートからメモリにロードし、実行します。

MERGE

機 能

プログラムを通信用ポートからロードし、メモリ上のプログラムとマージ（混合）します。

書 式

MERGE "COM[デバイス番号]："

解 説

アスキー形式のBASICプログラムを通信用デバイスからロードします。この時メモリ上にすでにあるプログラムはそのまま残り、MERGE文でロードされたプログラムと合併します。ただし、同じ行番号の場合はMERGEでロードされた方の行が残り、前のプログラムの行は消えます。

プログラム転送に際しエラーコレクションは行いません。

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。

文 例

MERGE "COM0:"

RUN

機 能

BASICプログラムを通信用ポートからメモリにロードし実行します。

書 式

RUN "COM[デバイス番号]："[,R]

解 説

アスキー形式のBASICプログラムをロードし実行します。RUN文実行前のプログラムを消去し、全てのファイルを閉じます。EOF（1AH）を受信するとプログラムのロードを終了し実行に入ります。Rオプションを指定すると、全てのファイルは開いたままに

なります。
プログラム転送に際しエラーコレクションは行いません。

デバイス番号
設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。

文 例

RUN "COM0:",R

BASICのプログラムをロードします。ファイルは閉じたままにします。

SAVE

機 能

BASICプログラムを通信用ポートに送ります。

書 式

SAVE "COM[デバイス番号]："

解 説

通信用ポートにアスキー形式のBASICプログラムを送出します。
プログラム転送に際しエラーコレクションは行いません。

デバイス番号
設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。

文 例

デバイス番号0にBASICのプログラムを送ります。

SAVE "COM0:"

3. ステートメント

CLOSE

機　能

OPEN 文で開いたファイルを閉じます。OPEN 文で開かれたファイルからデータを順に読み取り、変数に代入します

書　式

CLOSE[[#]ファイル番号][,[#]ファイル番号]...

解　説

OPEN 文で開かれたファイルの番号を指定して下さい。ファイルを閉じると、そのファイル番号が新たにファイルを開く時に使う事ができます。
指定したファイルが送信用に開かれたファイルであった場合は、相手機器に EOF コード（1AH）が送信されます。
RUN、END、CLEAR、NEW などのコマンドを実行した場合も、ファイルは閉じられます。

ファイル番号

対象となるファイル番号を 1〜(MAXFILES 文で指定された数）の整数値で指定します。省略時は全てのファイルを閉じます。指定には、整数型定数、変数、配列変数、式が使えます。

キャリア信号

CLOSE 文を実行すると回線に出ているキャリア信号はストップします。

文　例

CLOSE # 1,2,3

ファイル番号 1、2、3 のファイルを閉じます。

CLOSE

すべてのファイルを閉じます。

INPUT

機 能

OPEN 文で開かれたファイルからデータを順に読み取り、変数に代入します。

書 式

INPUT #ファイル番号,変数[,変数]...

解 説

受信バッファ用ファイルからデータを読み取り、変数に代入します。ファイル番号は、OPEN 文で受信バッファとして開かれたファイルを指定します。データが数値型の場合、データの前に書かれたスペース、リターンコード、ラインフィードコードは無視されます。データが文字型の場合、最初に読み取られた文字からスペース、カンマ、リターンコード、ラインフィードコードの前までを一つのデータとして読み取ります。" "で囲まれている場合は、" "の中の文字だけをデータとして読み取ります。変数を指定する場合は、読み取るデータに合ったタイプの変数を指定します。

ファイル番号

対象となるファイル番号を1～(MAXFILES 文で指定された数) の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

変数

数値型または文字型変数、それらの配列変数

文 例

```
10    OPEN"COM0:"FOR INPUT AS #1
20    IF EOF(1)THEN GOTO 50
30    INPUT #1,A$:PRINT A$
40    GOTO 20
50    CLOSE #1
```

ファイル番号を1として受信バッファ用ファイルを開きます。最後のデータが読み取られるまで順にデータを読み取って、文字型変数 A$に代入します。同時にデータを画面に表示します。データの読み取りが終了すると、ファイルを閉じます。

LINE INPUT

機　能

254文字までの文字列をファイルから読み取り、文字型変数に代入します。

書　式

LINE INPUT #ファイル番号,変数

解　説

受信バッファ用ファイルから文字型データを読み取りますが、INPUT #文と異なり、スペース、カンマ、ラインフィードコードを文字列データとして変数に代入します。データの区切とみなされるのはリターンコードだけです。ファイル番号は、OPEN文で受信バッファとして開かれたファイルを指定します。

ファイル番号

対象となるファイル番号を1〜(MAXFILES文で指定された数）の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

変数

文字型変数、配列変数です。

文　例

```
10      OPEN"COM0:"FOR INPUT AS #1
20      IF EOF(1)THEN GOTO 60
30      LINE INPUT #1,A$
40      PRINT A$
50      GOTO 20
60      CLOSE #1
```

ファイル番号を1として、受信バッファ用のファイルを開きます。

最後のデータが読み取られるまで文字型データを読み取り、文字型変数A$に代入して表示し、ファイルを閉じます。

OPEN

機　能

通信用ファイルを開きます。

書　式

OPEN ”COM[デバイス番号]：”[FOR モード] AS [#]ファイル番号

解　説

通信の入出力に必要な送信、受信バッファの領域をファイルとして確保します。OPEN文は次のコマンドや関数を使用する前に必ず実行してください。

PRINT #、PRINT # USING、LINE INPUT #、INPUT$

デバイス番号

設定の対象となるカートリッジを0（*）～9の整数値で指定します。指定には、文字型変数、変数、配列変数、式が使えます。

モード

オープン時のモードを指定します。

OUTPUT	送信モード
INPUT	受信モード
省略	OUTPUT、INPUT 両用（ファイルを閉じた時、EOF は送信されない）

ファイル番号

OPEN 文で受信バッファとして開かれたファイル番号を1～(MAXFILES 文で指定された数)の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

文　例

OPEN ''COM0:''FOR OUTPUT AS #1

送信バッファとしてファイル1を開きます。

PRINT

機　能

OPEN 文で開かれたファイルにデータを書き込みます。

書　式

PRINT #ファイル番号,[式][セパレータ][式]

解　説

式で指定したデータを送信バッファ用のファイルに書き込みます。

ファイル番号

OPEN 文で受信バッファとして開かれたファイル番号を1～(MAXFILES 文で指定された数)の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

式

文字型、数値型の定数、変数、配列変数、式が使えます。

セパレータ

カンマ (,)　　　データは14桁ごとに区切りをつけて書き込まれます。

セミコロン (;)　次のデータが直後に続いて書き込まれます。

また、正の数値データを書き込むと、データの区切りを示すスペースがデータの間に挿入されます。

末尾にセパレータを書かないと、データ送信後CRコード (0DH) とLFコード (0AH) が続いて書き込まれます。

文　例

```
10      OPEN"COM0:"FOR OUTPUT AS #1
20      A$="ABC":B$="DEF"
30      PRINT #1,A$'B$
40      PRINT #1,A$;B$
50      PRINT #1,+50,-50
60      CLOSE #1
```

送信バッファのファイル1に次の形式で順にデータを書き込みます。

```
 ABC            DEF CR/LF ABCDEF CR/LF
└──────────────┘
     14桁
 50             -50 CR/LF
└──────────────┘
     14桁
```

PRINT # USING

機　能

OPEN 文で開かれたファイルに指定した書式でデータを書き込みます。

書　式

PRINT # ファイル番号,USING 書式記号;式[,式]...

解　説

式の値が、書式記号で指定した形式で書き込まれます。

ファイル番号

OPEN 文で受信バッファとして開かれたファイル番号を1～(MAXFILES 文で指定された数) の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

式

文字型、数値型定数、変数、配列変数、式が使えます。

書式記号

記　号	表現形式と実行例
!	最初の1文字を出力します。 PRINT #1,USING "! " ; " United " ," Nation " 送信データ→UN
&　& n個の スペース	n+2文字を出力します。データがn+2文字より少ない場合は、残りの文字数だけスペースを挿入します。 PRINT #1,USING "& &" ; "ABCDEF" ,"GHI" ,"JKLM" 送信データ→ABCDGHI JKLM
@	文字列全部を出力します。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 A$ = "NORTH":B$ = "SOUTH" 30 PRINT #1,USING "@ POLE" ; A$,B$ 40 CLOSE #1 送信データ→NORTH POLE SOUTH POLE
#	送信したい数字の桁数だけ「#」を書きます。少数点は「.」です。 PRINT #1,USING " POINT:###.#" ; 1 2 3.4 送信データ→POINT:123.4 整数部分の桁数が指定した#の数より少ない場合、データは右詰めで表示され、多い場合はデータの前に%が付きます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING"####";12 30 PRINT #1,USING"####";12345 40 CLOSE #1 送信データ→12 %12345 数値データの少数部分の桁数が指定した#の数より少ない場合、0が追加され、多い場合は四捨五入されます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING"##.##";25.3 30 PRINT #1,USING"##.##";25.345 40 CLOSE #1 送信データ→25.30 25.35

記　号	表現形式と実行例
	数値データの「+」記号は無視、「−」記号は1桁として数えます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING "###";+123 30 PRINT #1,USING "###";−123 40 CLOSE #1 送信データ→123 %−123
+	数値データの前または後ろに、正数ならば「+」を、負数ならば「−」を付けます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING "+####";123,−123 30 PRINT #1,USING "####+";123,−123 40 CLOSE #1 送信データ→+123 −123 123+ 123−
−	負の数値データの後ろに「−」を付けます。 PRINT #1,USING "###−";123,−123 送信データ→123 123−
**	数値データのスペースを「*」で埋めます。書式中の「*」もひとつで1桁に数えられます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING "**######";123 30 PRINT #1,USING "**######";−234 40 CLOSE #1 送信データ→***** 123 **** −234
¥¥	数値データの前に「¥」を付けます。書式中の「¥」もひとつで1桁に数えられます。 10 OPEN "COM0:" FOR OUTPUT AS #1 20 PRINT #1,USING "¥¥###";1234 30 PRINT #1,USING "¥¥###";−1234 40 CLOSE #1 送信データ→¥1234 −¥1234
**¥	数値データの直前に「¥」を付け、それより前のスペースを「*」で埋めます。 PRINT #1,USING "**¥###.##";12.34 送信データ→***¥12.34
,	少数点の前のいずれかの箇所に「,」を付けると、少数点から左に3桁ごとにカンマを入れて出力します。 PRINT #1,USING "#,######.##";12345.67 送信データ→12,345.67
^^^^	数値データを浮動小数点で出力します。「^^^^」は指数部用のスペースに対応します。 PRINT #1,USING "##.##^^^^";234.56 送信データ→2.35E+02

4. 関数

EOF

機　能

ファイルの最後のデータが読み取られたら−1、それ以外なら０を返します。

書　式

EOF(ファイル番号)

解　説

OPEN 文で受信バッファとして開かれたファイルの番号を指定します。
この文により、受信バッファ内にある最後のデータが読み取られると−1を返します。それ以外なら０を返します。

ファイル番号

対象となるファイル番号を1〜(MAXFILES 文で指定された数）の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

返される値

整数値（−1または0)

文　例

```
IF EOF(1) THEN CLOSE #1
```

ファイル１の最後のデータが読み取られたら、そのファイルを閉じます。

INPUT $

機　能

ファイルから指定した数の文字を入力します。

書　式

INPUT$(文字数,[#]ファイル番号)

解　説

指定された数の文字(文字型データ)を受信バッファ用のファイルから入力します。ファイル番号は、OPEN 文で受信バッファとして開かれたファイルを指定します。

文字数

入力する文字数を1～255の整数値で指定します。指定には、数値型定数、変数、配列変数、式が使えます。

ファイル番号

対象となるファイル番号を1～(MAXFILES 文で指定された数) の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

返される値

文字型です。

文　例

```
10      OPEN"COM0:"FOR INPUT AS #1
20      X$=INPUT$(50,#1)
30      CLOSE
```

ファイル番号を1として受信バッファ用ファイルを開き、そのファイルから50文字入力したところで、それを文字変数X$に代入して、ファイルを閉じます。

LOC

機　能

ファイルの中の文字数を返します。

書　式

LOC(ファイル番号)

解　説

受信バッファ用ファイルの文字数を返します。

ファイル番号

OPEN 文で受信バッファとして開かれたファイル番号を1～(MAXFILES 文で指定された数)の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

返される値

0以上128以下の整数が返されます。

LOF

機　能

ファイルの中の空きスペースの大きさを返します。

書　式

LOF(ファイル番号)

解　説

受信バッファ中の空きスペースを文字数で返します。

ファイル番号

OPEN文で受信バッファとして開かれたファイル番号を1〜(MAXFILES文で指定された数)の整数値で指定します。指定には、整数型定数、変数、配列変数、式が使えます。

返される値

0以上128以下の整数が返されます。

2.6 拡張 BIOS

2.6.1 概要

MSX MODEM 拡張 BIOS は、MSX システムのモデム装置のソフトウェアインターフェイスを標準化して、アプリケーションソフトウェアがハードウェアに独立して作成できるようにするのが目的です。

モデム、ダイアラなどのハードウェアを標準化するのは、次々と新しいものが発表されている現状ではきわめて困難と考えられます。しかし、MSX システムの特長であるソフトウェアの互換性を実現するために BIOS でのインターフェイスを標準化しようとするものです。

この BIOS インターフェイスを守って作成されたソフトウェアは、たとえハードウェア仕様が異なるシステムでも互換性を持って使用することが可能となります。

MSX MODEM 拡張 BIOS は拡張 BIOS コールによりアプリケーションソフトウェアから自由に利用できます。BIOS の各ルーチンは、MSX MODEM 拡張 BIOS のジャンプテーブルを経由してインタースロットコールなどにより呼び出されます。

高速処理を必要とする場合は、あらかじめスロットをイネーブルしておき、直接コールすることもできます。この章では、MSX-RS232C 拡張 BIOS を使用するのに必要な、拡張 BIOS コールの方法と各 BIOS の機能について解説します。

アプリケーションは先ず拡張 BIOS コールによりジャンプテーブルのエントリを調べる必要があります。MSX MODEM 拡張 BIOS のデバイス番号は「8」です。これは RS-232C デバイスと同じデバイス番号となっており、RS-232C 用ソフトウェアとの互換性を保っています。

以下 MSX MODEM 拡張 BIOS に必要な拡張 BIOS コールについて解説します。

2.6.2 拡張 BIOS の呼び出し

1. ジャンプテーブルアドレスの取得

アプリケーションは、まず以下の拡張 BIOS コールにより、拡張 BIOS の存在するスロットとジャンプテーブルの先頭アドレスを調べなければなりません。

拡張 BIOS の存在するスロットとジャンプテーブルの先頭アドレスは、以下のようにして求めます。

1. RETURN 情報エリア用のワークエリア（64バイト）を取る。
2. 以下の設定を行い、0FFCAH 番地をコールする。

コール手順

D　　デバイス番号（8）
　　MSX MODEM 拡張 BIOS のデバイス番号は 8
E　　ファンクション番号（0）
B　　RETURN 情報エリアのスロットアドレス
　　スロットアドレスは、システムワークエリアに保存されている
HL　　RETURN 情報エリアの先頭アドレス

RAM のスロットアドレスは以下のワークエリアに保存されています。このワークエリアはディスクが接続されているシステムで有効です。

表 7.18　RAM のスロットアドレス

ページ	ワークエリアのアドレス
0	F341H
1	F342H
2	F343H
3	F344H

戻り値

B　　次の RETURN 情報エリアのスロットアドレス
HL　　次の RETURN 情報エリアの先頭アドレス

変更レジスタ

F

RETURN 情報はアプリケーションが指定した領域に次のように格納されます。

B:HL→	ジャンプテーブルのスロットアドレス
	ジャンプテーブルの先頭アドレス（下位）
	ジャンプテーブルの先頭アドレス（上位）
	リザーブ（常に0）
B:HL returned→	

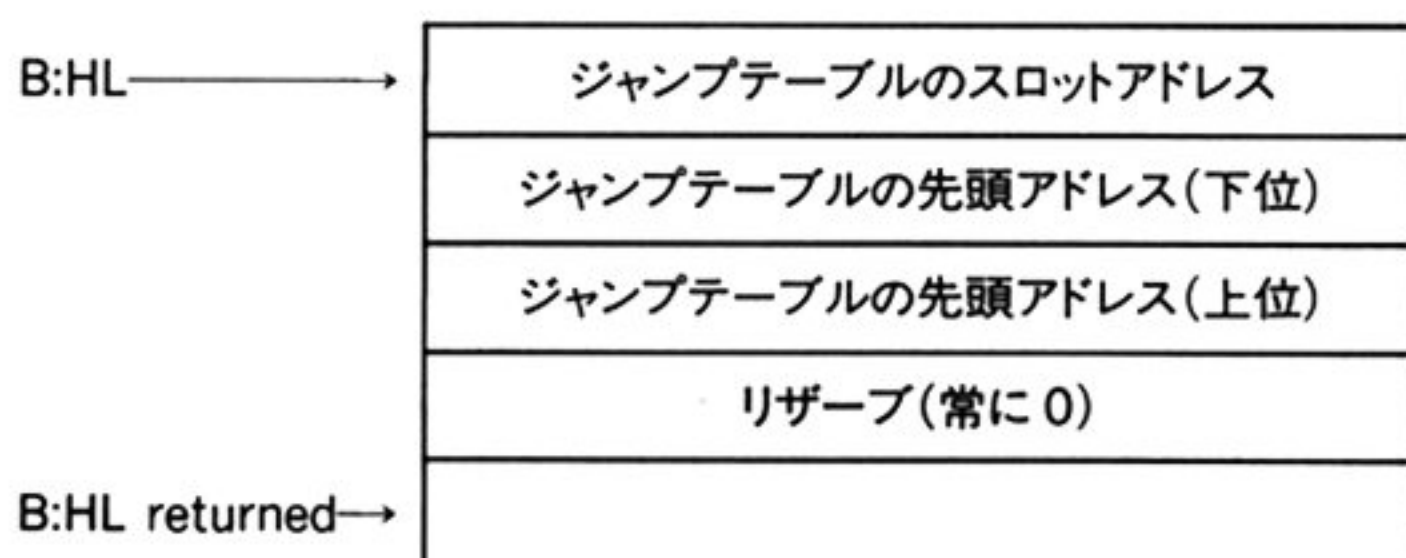



図 7.11　RETURN 情報の形式

MSX MODEMが無いときは、BレジスタとHLレジスタの内容が変わらずに返ってきます。スロットアドレスの表現はMSX共通で以下の通りです。

7	6	5	4	3	2	1	0
F	0	0	0	S	S	P	P

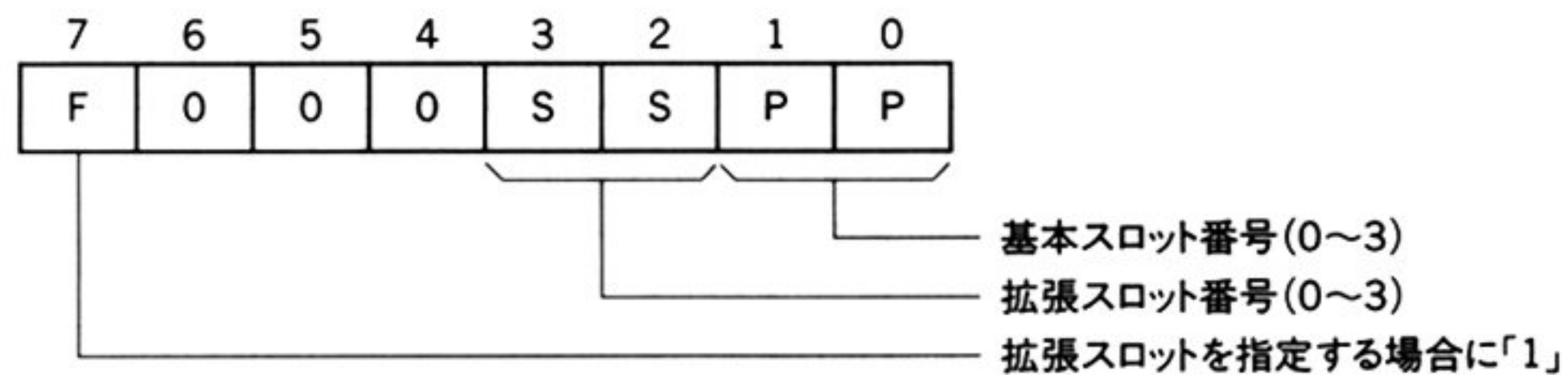



図7.12　スロットアドレスの形式

拡張BIOSを使用する場合は、この拡張BIOSコールで得られたジャンプテーブルをインタースロットコールなどにより呼び出し、目的のBIOSを使用します。

2. BIOSジャンプテーブル

拡張BIOSは以下に示すジャンプテーブルを持っています。アプリケーションソフトウェアは、インタースロットコールなどで各エントリを呼び出すことにより、BIOSの各機能を利用できます。

```
EXBTBL: DEFB    DVINFB,DVTYPE,0   ; Device information and Device type
        JP      INIT              ; Initialize MODEM port
        JP      OPEN              ; Open MODEM port
        JP      STAT              ; Read status
        JP      GETCHR            ; Receive data
        JP      SNDCHR            ; Send data
        JP      CLOSE             ; Close MODEM port
        JP      EOF               ; EOF code received
        JP      LOC               ; Reports the number of characters
                                  ; in the receiver buffer
        JP      LOF               ; Reports the number of free spaces
        z                         ; left in the receiver buffer
        JP      BACKUP            ; Back up a character
        JP      SNDBRK            ; Send break character
        NOENT                     ; Reserved for RS-232C BIOS
        NOENT                     ; Reserved for MULTI RS-232C BIOS
        JP      NCUSTA            ; NCU status
        JP      SPKCNT            ; Speaker control
        JP      LINSEL            ; Select line for TEL or MODEM
        JP      DIALST            ; Put dial string
        JP      DIALCH            ; Put dial character
        JP      DTMFST            ; DTMF Status
        JP      RDDTMF            ; Read DTMF signal
        JP      HOKCNT            ; Hook control
        JP      CONFIG            ; Device configration
        JP      SPCIAL            ; Special Control
        NOENT                     ; Reserved for future expansion
        NOENT
```

■ DVINFB

デバイスインフォメーションバイト(DVINFB)はMODEMカートリッジに、各オプションがあるかどうかを示します。

7	6	5	4	3	2	1	0
0	0						0

- ビット0: 予約
- ビット1: 送信割り込み
- ビット2: BREAKキャラクタ検出
- ビット3: タイマインタラプト
- ビット4: キャリアデテクト(CD)
- ビット5: リングインジケータ(RI)
- ビット6: 予約
- ビット7: 予約

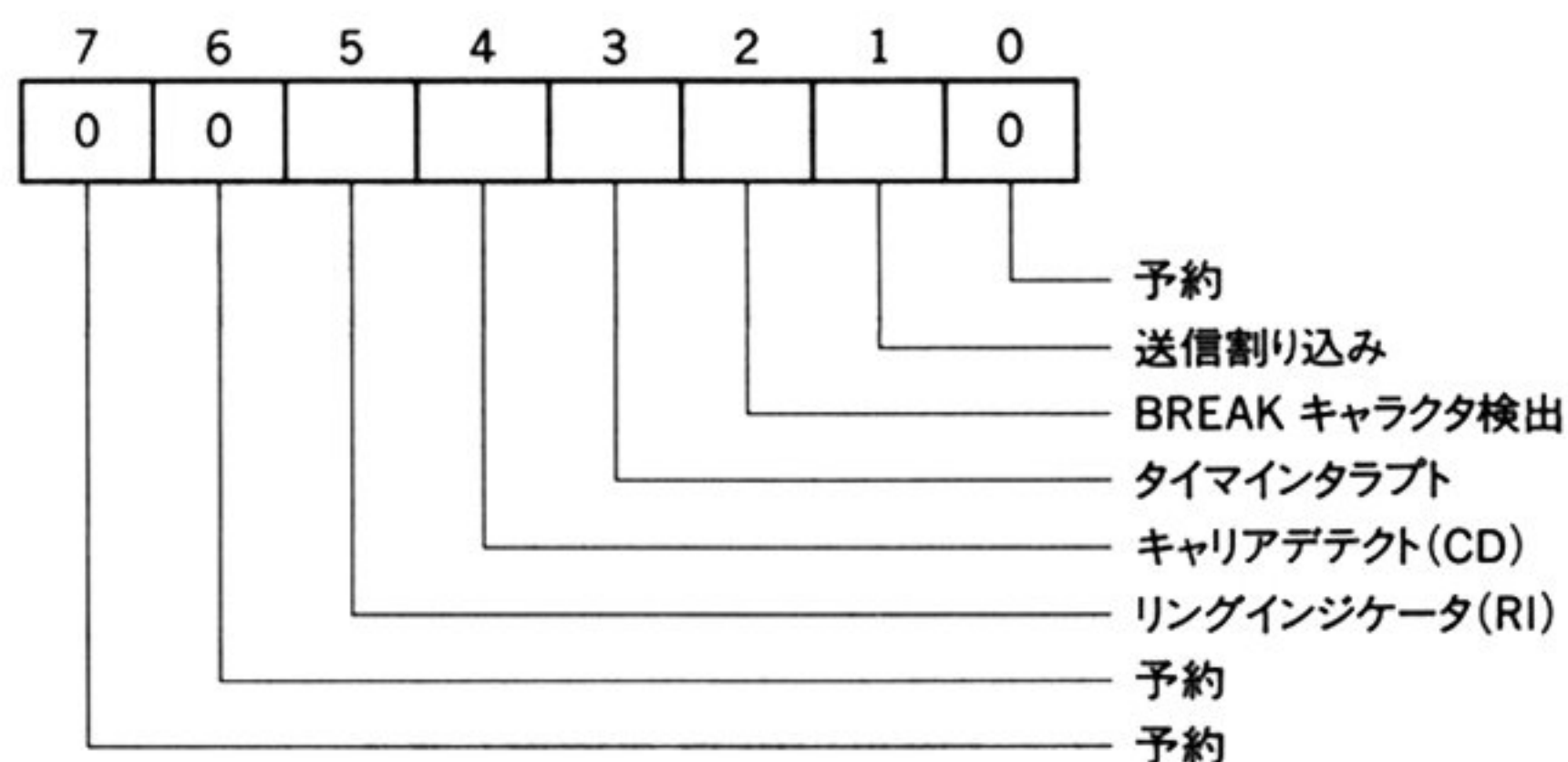



図7.13　DVINFBの形式

各ビットは1でその機能があることを示し、0でないことを示します。

■ DVTYPE

デバイスタイプバイト（DVTYPE）はこの装置がマルチタイプかシングルタイプかを示します。マルチタイプとは複数のMSX MODEMカートリッジを同時にスロットに挿入でき、個々に制御できる装置のことです。

DVTYPEが0以外であれば、マルチタイプのモデムカートリッジです。

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	

- ビット0: マルチタイプ
- ビット1〜7: 予約

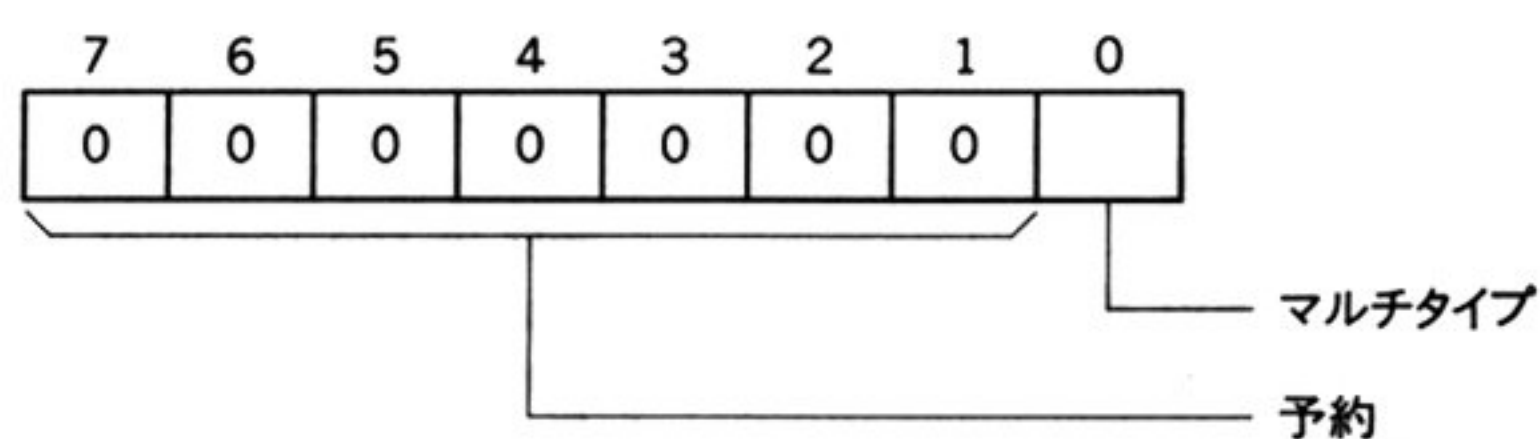



図7.14　DEVTYPEの形式

3. 拡張 BIOS の各機能

拡張 BIOS の各機能はインタースロットコールにより呼び出されます。インタースロットコール（CALSLT）はの呼び出しアドレスは 001CH です。

コール手順

IY	上位 8 ビットにスロットアドレス
IX	コールアドレス
	その他のレジスタは機能により異なります。各項を参照してください。

INIT

機　能

モデムポートをイニシャライズします。

コール手順

AF　MODEMの種類

0	Bell	103	300bps	Full	duplex
1	Bell	212A	1200bps	Full	duplex
2	CCITT	V21	300bps	Full	duplex
3	CCITT	V22	1200bps	Full	duplex
4	CCITT	V22bis	2400bps	Full	duplex
5	CCITT	V23	1200bps	Half	duplex
6	CCITT	V27ter	4800bps	Half	duplex
7	CCITT	V29	9600bps	Half	duplex
8	CCITT	V32	9600bps	Full	duplex
9〜254	システム予約				
255	初期設定値				

B　パラメータテーブルのスロットアドレス

C　ダイヤラのモード

0	プッシュボタン（DTMF）
1	システム予約
2	ダイヤルパルス（20pps）
3	ダイヤルパルス（10pps）
4	自動検出
5〜254	システム予約
255	初期設定値

HL　パラメータテーブルのアドレス

B:HL ⟶ (+0)

オフセット	意味	
+0	キャラクタ長	5〜8
+1	パリティ	E、O、I、N
+2	ストップビット	1、2、3
+3	XON / XOFF 制御	X、N
+4	RS / CS 制御	H、N
+5	受信 LF 挿入	A、N
+6	送信 LF 削除	A、N
+7	SI / SO 制御	S、N
+8	受信速度 50〜19200	(Low) (High)
+10	送信速度 50〜19200	(Low) (High)
+12	送信タイムアウト	0〜255

図 7.15　INIT パラメータテーブルの形式

オフセット＋0 から＋7 はアスキーキャラクタ、＋8 から＋12 まではバイナリです。
アスキーキャラクタはすべて大文字とします。

戻り値

パラメータの指定に間違いがある場合にキャリーフラグが 1 にセットされます。

変更レジスタ

AF

解　説

このエントリは他の機能を呼び出す時に、必ず呼び出されなくてはなりません。パラメータは拡張 BASIC の COMINI と同等です。
また、A レジスタにモデムの種類を、C レジスタにダイヤラのモードを設定します。
モデムの種類と送受信速度との関係は以下の順でチェックします。
モデムの種類が正しく示されている場合は、その送受信速度にしたがい「Receiver Band Rate」および「Transmitter Band Rate」はチェックされません。モデムの種類がデフォルトの場合に限り、「Receiver Band Rate」および「Transmitter Band Rate」がチェックされます。この指定が正しく行われていれば、それに対応したモデムが選択されます。
指定が正しくない場合はデフォルトのモデムが選択されます。
初期設定値が指定された場合は、DIP スイッチやメモリスイッチで指定されるもの、ま

たはシステムであらかじめ決められているものを使用します。
ダイヤラの自動検出モードとは、トーン検出機能を使い、次のように動作するものをいいます。

1. まず、DTMFにて最初のダイヤルトーンを発信します。この時、回線上のトーンが消失すれば、この回線はDTMFを受け付けるものとしてダイヤルを続けます。
2. もし、トーンが持続している場合には、パルスダイヤルモードにし、10ppsでダイヤルします。

OPEN

機　能

MODEMポートをオープンします。

コール手順

HL	FCBのアドレス（8000H番地以上のアドレスを指定）
C	バッファ長（32～254）
E	オープンモード

1	INPUTモード
2	OUTPUTモード
4	RAW and INPUT/OUTPUTモード

FCBの取り方

HL→	9bytes for File Control
	[C]×2 Bytes Receiver Buffer

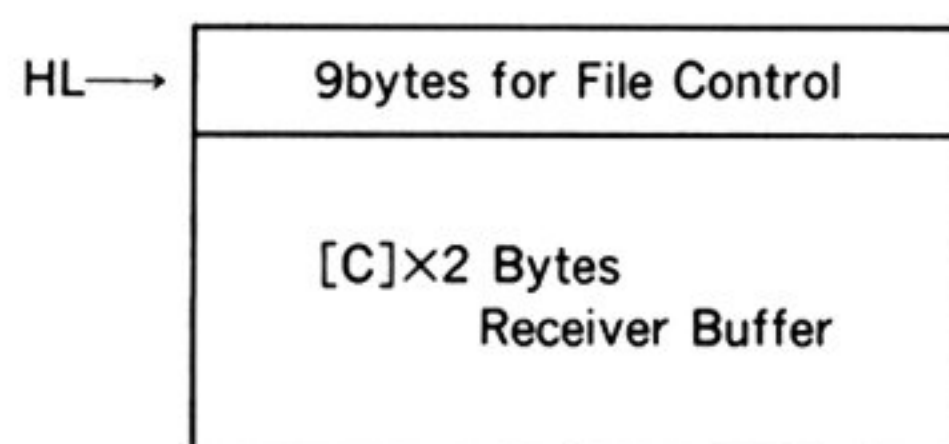



図7.16　FCBの形式

戻り値

オープンモードが正しくない場合はキャリーフラグがセットされます。

変更レジスタ

AF

解　説

オープンするためには、ファイルコントロールブロック（FCB）を設定する必要があります。この機能はMODEMのすべてのI/O処理に先立って行う必要があります。
FCBエリアは9バイトのワークエリアと受信するデータの2倍のバッファが必要です。受信データは受信データそのものと、エラーステイタスの2バイト構成でバッファに格納されます。このバッファ長はCレジスタで指定されます。

STAT

機　能

ハードウェアの状態を返します。

コール手順

なし

戻り値

HL　ステータス

ビット0	CD信号
ビット1	RI信号
ビット2	ブレーク信号検出
ビット3	DR信号（常にready）
ビット4	システム予約
ビット5	システム予約
ビット6	システム予約
ビット7	システム予約
ビット8	システム予約
ビット9	ロングループアラーム
ビット10	CTRL + STOP キー検出
ビット11	パリティエラー
ビット12	オーバーランエラー
ビット13	フレーミングエラー
ビット14	タイムアウトエラー
ビット15	バッファオーバーフロー

ビット0からビット7はLレジスタを、ビット8からビット15はHレジスタを指します。サポートされていない機能のビットは0を返します。

変更レジスタ

HL

GETCHR

機 能

受信バッファから文字を読み出します。

コール手順

なし

戻り値

A　　受信文字
　　エラーがある場合はサインフラグがセットされます。
　　文字がEOFコードである場合はキャリーフラグがセットされます。

変更レジスタ

F

SNDCHR

機 能

MODEMに文字を送出します。

コール手順

A　　送信文字

戻り値

[CTRL]+[STOP]が押されているとキャリーフラグがセットされます。XON待ち状態でタイムアウトが発生した場合にゼロフラグがセットされます。

変更レジスタ

F

解　説

イニシャライズ時にXON/XOFFコントロールが指定されていれば、自動的にこの制御が行われます。また、タイムアウトが指定されていれば同様に制御されます。

CLOSE

機　能

MODEMポートをクローズします。

コール手順

なし

戻り値

エラー発生時にキャリーフラグがセットされます。

変更レジスタ

AF

解　説

FCBが解放され、このポートがOUTPUTモードの指定がされている時は、EOFコードが送出されます。

EOF

機　能

次の文字がEOFかどうかをチェックします。

コール手順

なし

戻り値

HL	−1	EOFの場合にキャリーフラグがセットされます。
	0	EOFでない場合にはキャリーフラグがリセットされます。

変更レジスタ

AF

LOC

機　能

受信バッファの中にある文字数を返します。

コール手順

なし

戻り値

HL　　受信バッファ内の文字数

変更レジスタ

AF

解　説

この機能が返す値にはBACKED-UP文字が含まれます。デバイスがINPUTモードでオープンされている場合には、EOF以降の文字は無視されます。ただし、この文字もバッファのスペースを必要とします。

LOF

機　能

受信バッファの空き領域を返します。

コール手順

なし

戻り値

HL　　受信バッファの空き領域

変更レジスタ

AF

BACKUP

機　能

文字を受信バッファに戻します。これは1文字のみ可能です。

コール手順

C　　バックアップする文字

戻り値

なし

変更レジスタ

AF

SNDBRK

機　能

指定した数のブレーク文字を送出します。

コール手順

DE　　送出するブレーク文字数

戻り値

[CTRL] + [STOP] が押されるとキャリーフラグがセットされます。

変更レジスタ

AF、DE

解　説

ブレーク送出中に [CTRL] + [STOP] が押されると、キャリーフラグをセットしてアボートリターンします。

NCUSTA

機　能

NCUのハードウェアの状態を返します。

コール手順

なし

戻り値

HL　ステータス

L　ビット0　Ring indicator (RI信号がある間、1)

ビット1
ビット2　Dialer mode

ビット2	ビット1	
0	0	プッシュボタン (DTMF)
0	1	自動検出
1	0	ダイヤルパルス (20pps)
1	1	ダイヤルパルス (10pps)

ビット3
ビット4　回線極性検出

ビット4(LA)	ビット3(LB)	
0	0	loop off
0	1	DC loop
1	0	DC loop
1	1	予約

ビット5　課金パルス検出 (極性反転をラッチする。リードセット)

ビット6　Call progress tone 検出 (400Hzのトーンを検出すると1)

ビット7　外部電話機の HOOK 状態検出

0	ON HOOK
1	OFF HOOK

H　0

サポートされていない機能のビットは0またはunknownを返します。

変更レジスタ

HL

SPKCNT

機　能

スピーカをON/OFFします。

コール手順

AF	0	スピーカOFF
	0以外	スピーカON

戻り値

サポートされていない場合、キャリーフラグがセットされます。

変更レジスタ

なし

LINSEL

機　能

電話回線を電話器に接続するか、内蔵モデム・ダイヤラに接続するかを選択します。

コール手順

AF	ビット0	回線を外部電話機に接続します	
		1	電話
		0	モデム
	ビット1〜2	内蔵ハンドセットを回線に接続します。	
		ビット1	マイク
		ビット2	スピーカ
	ビット3〜4	内蔵ハンズフリーフォンを回線に接続します。	
		ビット3	マイク
		ビット4	スピーカ
	ビット5〜7	予約	

戻り値

パラメータにエラーがある場合にキャリーフラグがセットされます。

変更レジスタ

なし

DIALST

機　能

電話番号を送出します。

コール手順

B　　ダイヤルパラメータ列のスロットアドレス

HL　　ダイヤルパラメータ列のアドレス

ダイヤルデータ

0〜9、A〜D、H、#、＊、<、：、T

データの最後（ターミネータ）は00H

H	約1秒間、ON HOOKとします。
<	3秒間のポーズをつくります。
：	第2ダイアルトーンを待ちます。トーン検出の機能のない場合は、「<」に読み換えます。
T	トーンダイヤルへの切り換えを行います。

上記のパラメータのなかでサポートしていないものが指定された場合は、エラーを返します（キャリーフラグをセット）。上記以外のデータは無視されます。

C

0	プッシュボタン（DTMF）
1	システム予約
2	ダイヤルパルス（20pps）
3	ダイヤルパルス（10pps）
4	自動検出
5〜254	システム予約
255	初期設定値

初期設定値についてはINITを参照して下さい。

戻り値

パラメータにエラーがある場合はキャリーフラグをセットします。

変更レジスタ

なし

解　説

電話回線をダイヤラに接続し、3秒後に(ただし、Call progress 検出機能がある場合は、第1ダイヤルトーン検出後すみやかに) ダイヤル送出を開始します。ダイヤル終了後、回線はモデム側に接続されたままです。

DIALCH

機　能

1桁だけダイヤル信号を送出します。電話回線の切り換えは行いません。

コール手順

AF　ダイヤル文字

　　ダイヤルデータの取り扱いはDIALST に準じます。

C

0	プッシュボタン (DTMF)
1	システム予約
2	ダイヤルパルス (20pps)
3	ダイヤルパルス (10pps)
4〜254	システム予約
255	初期設定値

　　初期設定値についてはINIT を参照して下さい。

戻り値

パラメータにエラーがある場合はキャリーフラグをセットします。

変更レジスタ

なし

DTMFST

機　能

DTMF デコーダのステータスを調べます。

コール手順

なし

戻り値

DTMF コードが入力されていればゼロフラグがセットされます。この機能がサポートされていなければキャリーフラグがセットされます。

変更レジスタ

AF

RDDTMF

機 能

DTMF デコーダよりデータを読み取ります。DTMF 信号が無い場合は新しい信号があるまで待ちます。

コール手順

なし

戻り値

A　　DTMF コード（アスキーコード）
　　 [CTRL] + [STOP] が押されたか、この機能がサポートされていなければキャリーフラグをセットします。

変更レジスタ

AF

HOKCNT

機 能

回線の切断・接続を行います。

コール手順

AF	1	OFF HOOK
	0	ON HOOK

戻り値

この機能がサポートされていなければキャリーフラグをセットします。

変更レジスタ

なし

CONFIG

機　能

システムのハードウェアのコンフィギュレーションを返します。サポートしているものについて対応するビットを1にします。

コール手順

A	0	MODEM 機能

戻り値

HL	ビット0	Bell	103	300bps	Full	duplex
	ビット1	Bell	212A	1200bps	Full	duplex
	ビット2	CCITT	V21	300bps	Full	duplex
	ビット3	CCITT	V22	1200bps	Full	duplex
	ビット4	CCITT	V22bis	2400bps	Full	duplex
	ビット5	CCITT	V23	1200bps	Half	duplex
	ビット6	CCITT	V27ter	4800bps	Half	duplex
	ビット7	CCITT	V29	9600bps	Half	duplex
	ビット8	CCITT	V32	9600bps	Full	duplex
	ビット9〜15	システム予約				

コール手順

A	1	DIALER 機能

戻り値

HL	ビット0	プッシュボタン（DTMF）
	ビット1	ダイヤルパルス（20pps）
	ビット2	ダイヤルパルス（10pps）
	ビット3	自動検出
	ビット4	A〜Dをサポートしている。
	ビット5	Hをサポートしている。
	ビット6	プッシュボタン（DTMF）とダイヤルパルスの切り換えがソフトウェアによりできる。
	ビット7	10ppsと20ppsの切り換えがソフトウェアによりできる。
	ビット8〜15	システム予約

コール手順

A　　2　　LINE機能

戻り値

HL	ビット0	外部電話機
	ビット1	内蔵モデム
	ビット2	内蔵ハンドセット
	ビット3	内蔵ハンズフリーフォン
	ビット4	外部AUX入力
	ビット5〜15	システム予約

コール手順

A　　3　　その他の機能

戻り値

HL	ビット0	RI	1
	ビット1	Call progress検出	1
	ビット2	回線極性検出	1
	ビット3	課金パルス検出	0
	ビット4	DTMFデコーダ	0
	ビット5	スピーカ（sound in）	1
	ビット6	ON/OFF HOOK機能	1
	ビット7	外部電話機　ON/OFF HOOK検出機能	0
	ビット8	MSX標準カートリッジ	1
	ビット9	RS-232C（*）	0

ビット10	送出電力切替機能	1
ビット11	キャリア制御機能	0
ビット12	クオリティ検出機能	1
ビット13	イコライザ制御機能	1
ビット14〜15	システム予約	

(*) RS-232Cとモデム機能と切り換えて使用できるカートリッジの場合に1とする（例えば、同じUARTを使用して切り換えて使っているような場合)。

コール手順

A　4〜255　システム予約

戻り値

HL　0000H

変更レジスタ

HL

SPCIAL

機　能

メーカー別および機種別の特別な処理を行います。

コール手順

A　0　モデムの送出電力切換機能
C　ー送出電力値（dBm）
　255（初期設定値）

戻り値

この機能がサポートされていなければキャリーフラグをセットします。

コール手順

A　1　キャリア制御
C　0　キャリアOFF
　1　キャリアON
H　RS ONまでのディレイタイム（n×10mS）
L　CS ONからreturnまでの初期設定値（n×10mS）

戻り値

この機能がサポートされていなければキャリーフラグをセットします（全2重モデムではキャリアONが、半2重モデムではキャリアOFFか初期設定値となります）。

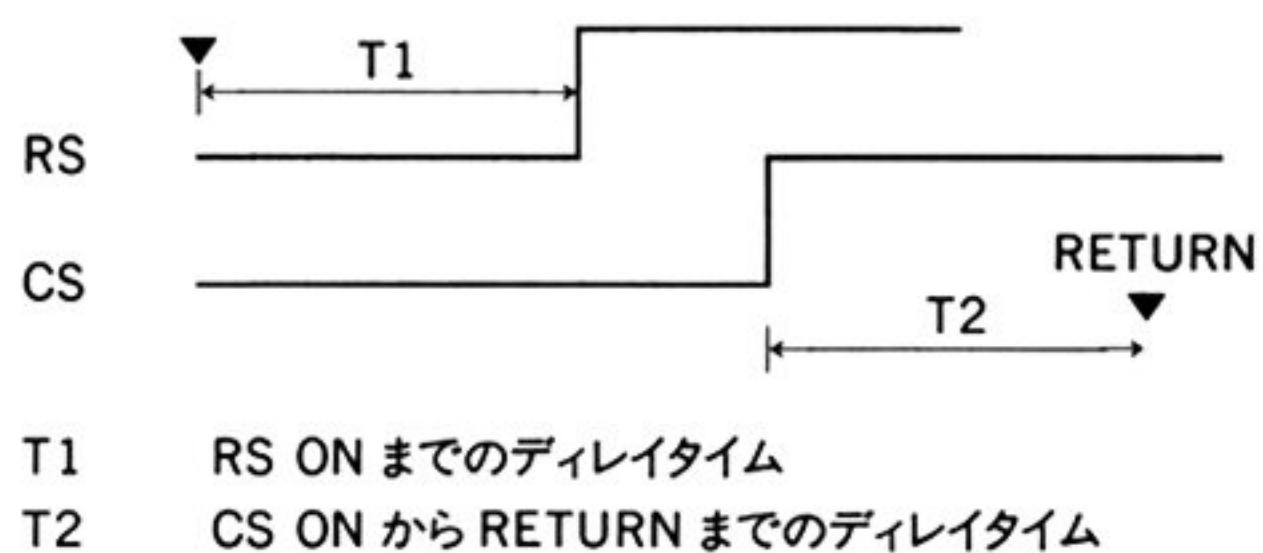



図7.17　キャリアONの制御

コール手順

A	0	イコライザを使用しない
	1	イコライザを使用する
	2	自動的にイコライザを調整する
	255	初期設定値

戻り値

この機能がサポートされていなければキャリーフラグをセットします。

3章 MSX-MUSIC

3.1 ハードウェア

3.1.1 メモリの構成

MSX BUS

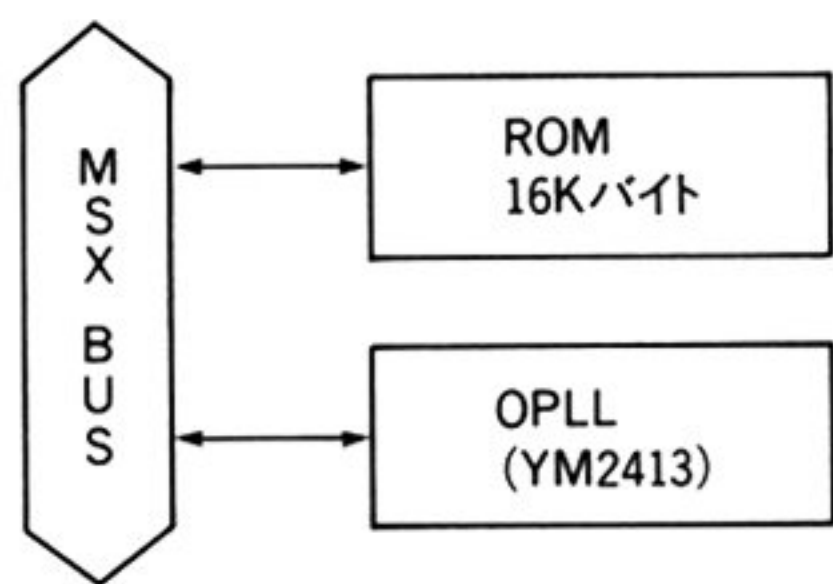
ROM
16Kバイト

OPLL
(YM2413)

図8.18　MSX-MUSICのメモリ構成

図 7.18　MSX-MUSIC のメモリ構成

3.1.2 I/O の構成

1. 本体内蔵の場合

表 7.19　MSX-MUSIC I/O 構成（本体内蔵）

I/O アドレス	Read / Write	OPLL レジスタ名
7CH	W	アドレスレジスタ
7DH	W	データレジスタ

2. 外付けの場合

パナアミューズメントカートリッジ2準拠

表7.20 MSX-MUSIC I/O構成(外付け)

I/Oアドレス	メモリアドレス	Read / Write	OPLLレジスタ名
7CH	7FF4H	W	アドレスレジスタ
7DH	7FF5H	W	データレジスタ

3.1.3 内蔵・外付け判別データ

表7.21 内蔵・外付け判別データ

アドレス	データ(内蔵)	データ(外付け)	
4018H	41H 'A'	50H 'P'	「PAC2」の文字列は「パナアミューズメントカートリッジ2」の場合。 この4バイトの文字列は、メーカーごとに異なる任意の文字列でかまわない。
4019H	50H 'P'	41H 'A'	
401AH	52H 'R'	43H 'C'	
401BH	4CH 'L'	32H '2'	
401CH	4FH 'O'	4FH 'O'	
401DH	50H 'P'	50H 'P'	
401EH	4CH 'L'	4CH 'L'	
401FH	4CH 'L'	4CH 'L'	

MSX-MUSICはオプション機能ですから、カートリッジを外付けすることにより、あとからFM音源を追加することができます。しかし、MSX-MUSICを内蔵したシステムにFM音源カートリッジを外付けすると、ひとつのシステムに同一アドレスのFM音源LSIが2つ存在することになり、音量が通常の2倍の大きさになってしまいます。

これを避けるために、外付けのカートリッジに内蔵されるソフトウェアは、そのカートリッジ内の音源LSIの動作を許可するかどうかを4018Hからの「APRLOPLL」文字列の有無を調べ、文字列がなければカートリッジのFM音源をI/Oポートに接続します。

3.1.4 OPLLを直接アクセスする場合の注意

この章で公開された内容を利用すれば、I/Oポートから OPLL を直接アクセスして音を鳴らすことができます。しかし、OPLL のレジスタへデータを書き込む場合、タイミングによっては、正常に動作しなくなる可能性があります。したがって、商用のアプリケーションソフトウェアは直接 OPLL をアクセスしてはいけません。OPLL にアクセスする場合は、必ず、FM BIOS を使用するようにして下さい。

1. ウエイト

OPLL では、内部レジスタにアドレスやデータを書き込むと、次の動作に移るまでにはウエイト時間が必要です。このウエイト時間は、レジスタのアドレスの指定とデータの書き込みで異なります。表 7.22 で指定された時間だけ、CPU は OPLL に対して、次の動作を待たなければなりません。

このウエイト時間を無視した場合は、そのときに設定したデータは保証されません。

表 7.22　OPLL のウエイト時間

モード	ウエイト時間
アドレス指定	3.36μsec
データ書き込み	23.52μsec

2. 内蔵と外付けの両方の音源に対応する方法

現在、市販されている外付けの MSX-MUSIC カートリッジとしては、「パナアミューズメントカートリッジ 2(以下、FM-PAC)」があります。FM-PAC には、MSX-MUSIC 内蔵マシンに装着されたときのことを考慮して、FM-PAC 内の音源 IC を ON、OFF できるようになっています(「3.1.3 内蔵・外付け判別データ」参照)。

MSX に電源が入ったときは、FM-PAC 内蔵音源 IC は OFF になっていて、BASIC で CALL MUSIC 命令が実行されたり、FM-BIOS の初期化ルーチンが呼ばれたりしたときに、もし MSX 本体に音源 IC が内蔵されていなければ ON されます。

したがって、MSX-MUSIC が内蔵されていない機種では、ユーザープログラムが直接 OPLL をアクセスする場合、以下の方法で FM-PAC 内蔵の音源 IC を ON しなければなりません。

1. 04018H 番地から 0401FH 番地に次の内容を持っているスロットを探します。

```
DB      "APRLOPLL"
```

2. もしあれば、MSX本体に音源ICが内蔵されているので、そのままで音源ICは操作できます。
3. ない場合には、0401CH番地から401FH番地に次の内容を持っているスロットを探します。

   DB ”OPLL”
4. もしなければFM音源IC自体が装着されていませんので、使用できません。
5. もしあればそのスロットの07FF6H番地のビット0を「1」にすることにより、FM音源ICをONできます。このときには、07FF6H番地の内容を1度読み出して、ビット0のみ「1」にして書き込まなければならないことに注意して下さい。

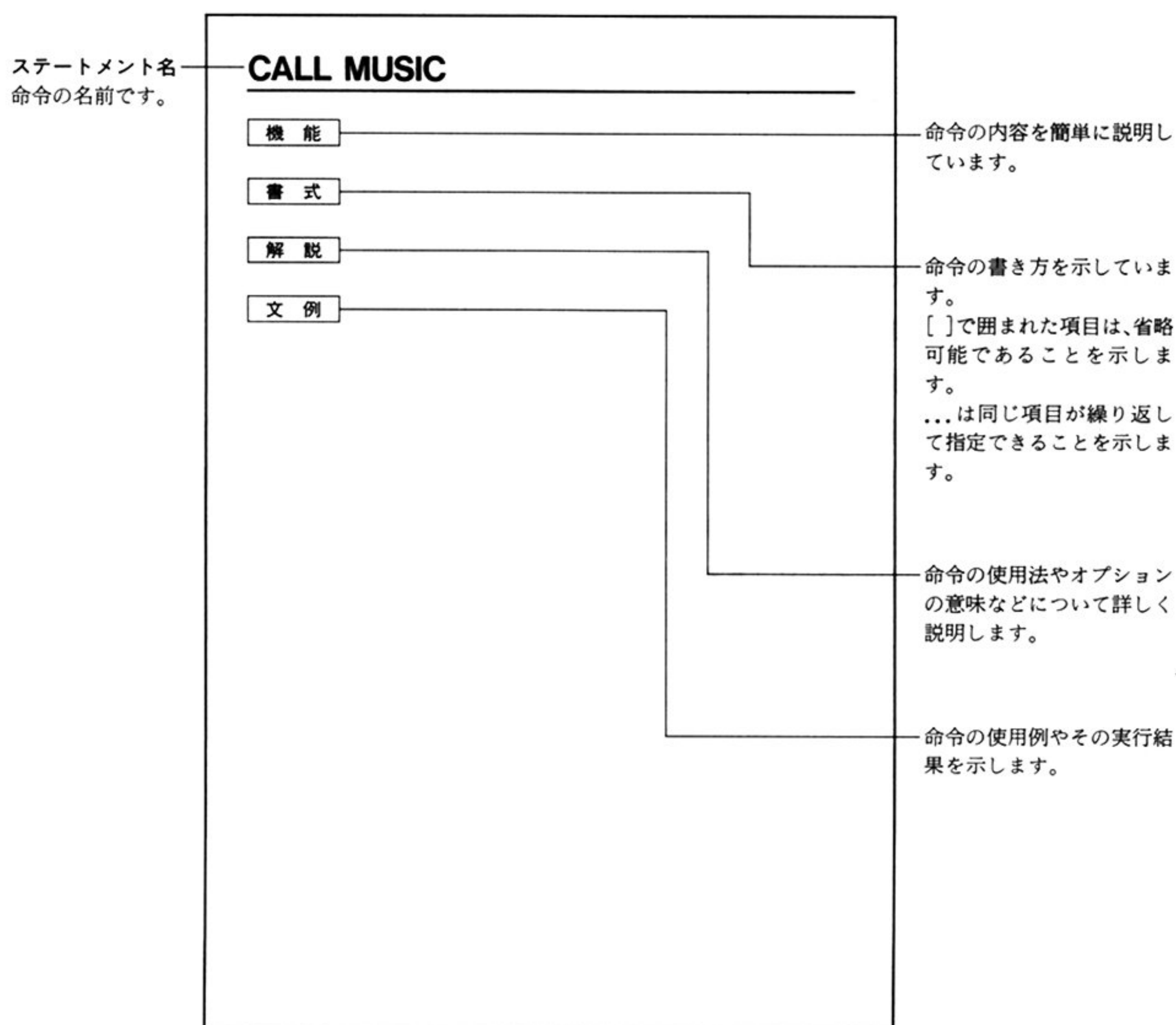
3.2 拡張BASIC

3.2.1 概要

MSX-MUSICには、各機能を簡単に使用できるように、MSX-MUSIC拡張BASICが用意されています。使い方はCALL MUSICのように拡張ステートメントの形式です。CALLは_（アンダーバー）で代用できます。

MSX-MUSICの拡張BASICはMSX-AUDIO拡張BASICのサブセット版です。MSX-AUDIO用のPCM関連のコマンドなどは使用できません。また、音色もできるだけMSX-AUDIOの音色に近づけてありますが、多少異なりますし、ユーザー音色の制限(MSX-MUSICでは1音色だが、MSX-AUDIOでは9音色）もあります。使えないコマンドについては、「3.2.5 MSX-AUDIOのステートメント」を参照して下さい。

3.2.2 この章の表記法

ステートメント名
命令の名前です。
CALL MUSIC
機能
命令の内容を簡単に説明しています。
書式
命令の書き方を示しています。
[ ]で囲まれた項目は、省略可能であることを示します。
...は同じ項目が繰り返して指定できることを示します。
解説
命令の使用法やオプションの意味などについて詳しく説明します。
文例
命令の使用例やその実行結果を示します。

3.2.3 拡張 BASIC コマンド一覧

1. 拡張ステートメント（CALL 文と共に使用します）

コマンド名	機　能	ページ
AUDREG	音源 LSI のレジスタに値を書き込みます。	228
BGM	バックグラウンド処理を行うかどうかを指定します。	228
MUSIC	MSX-MUSIC システムを初期化します。	229
PITCH	FM 音源の楽音の音高（ピッチ）を与えます。	230
PLAY	音楽をミュージックマクロランゲージにしたがって演奏します。	231
PLAY	PLAY 文が音楽を演奏中かどうかを返します。	235
STOPM	バックグラウンドで実行中の PLAY 文の演奏を停止します。	235
TEMPER	音律（テンペラメント）を与えます。	236
TRANSPOSE	FM 音源の楽音に対してセント単位で移調を与えます。	237
VOICE	FM 音源の各チャンネルに音色（ボイス）を直接に設定します。	237
VOICE COPY	音色パラメータデータの転送を行ないます。	242

3.2.4 拡張 BASIC の解説

CALL AUDREG

機 能

音源 LSI のレジスタに値を書き込みます。

書 式

CALL AUDREG(<レジスタ番号>, <値>[, <チャンネル番号>])

解 説

音源 LSI のレジスタに対する書き込みを行ないます。システムソフトウエアが割り込みなどでひんぱんに書き込んでいるレジスタには、効果がない場合やシステムの再立ち上げが必要な場合があります。

<チャンネル番号>は 0、または省略しなければなりません。これは MSX-AUDIO 拡張 BASIC との互換性をとるために用意されているのもので MSX-MUSIC では意味を持ちません。

文 例

```
CALL AUDREG(&H20,0)
```

CALL BGM

機 能

バックグラウンド処理を行うかどうかを指定します。

書 式

CALL BGM(<変数>)

解 説

<変数>は 0 または 1 の値をとり、次のような意味を持ちます。

0	PLAY 文のバックグラウンド処理を行なわない。
1	PLAY 文のバックグラウンド処理を行なう。

MUSIC 文による初期化ではバックグラウンド処理が指定されていますが、<変数>に

0を与えることでフォアグラウンド処理とすることができます。

文 例

CALL BGM(0)

PLAY 文のバックグラウンド処理を行なわない。

CALL BGM(1)

PLAY 文のバックグラウンド処理を行なう。

CALL MUSIC

機 能

MSX-MUSIC システムを初期化します。

書 式

CALL MUSIC[([<モード>][,[0][,<PLAY 文第1文字列へのチャンネル数>[,<PLAY 文第2文字列へのチャンネル数>[,･･･[,<PLAY 文第9文字列へのチャンネル数>]]]]]]]]]])]

解 説

音源LSIの初期化とともに9個のFM音源のチャンネルをどのように使用するかを指定します。MUSIC文により初期化を行うまでは、全ての拡張BASICステートメントは使用できません。<モード>は0か1で、0はリズム音を使用しないモードで、1はリズム音を使用するモードです。リズム音を使用する場合にはチャンネル7、8、9を使用しますので、楽音に使えるのは残り6チャンネルになります。したがって、PLAY文で使用するチャンネル数はリズム使用時には6以下、リズムを使わないときは9以下である必要があります。

チャンネルの使用割り当ては、PLAY文では、チャンネル番号の小さい方（1,2,3・・・）から割り当てます。

パラメータを1つ以上指定した場合、他のパラメータの省略時の値は0となります。

PLAY文の文字列へのチャンネル数を、0に設定したり省略したりすることはできません。次の例を参照して下さい。

```
CALL MUSIC(0,0,0,5,0)
               ↑ ↑
```

0を設定してはいけない（Illegal function call になる）

CALL MUSIC(0,0,1, ,2)

↑

省略してはいけない（Syntax error になる）

パラメータなしで使われたときは、

CALL MUSIC(1,0,1,1,1)

と同じになります。すなわち、

- FM 音源のチャンネル1を PLAY 文の最初の文字列に割り当てる。
- FM 音源のチャンネル2を PLAY 文の2番目の文字列に割り当てる。
- FM 音源のチャンネル3を PLAY 文の3番目の文字列に割り当てる。
- リズム音を PLAY 文の4番目の文字列に割り当て使用する。
- PLAY 文の5番目以降7番目までの文字列は PSG 音源の制御に割り当てる。

という意味になります。

MUSIC 文を実行すると、システムの割り込みのフックが MSX-MUSIC のシステムソフトウェアにリンクされるので割り込み処理ルーチンのオーバーヘッドが増え、システムのスループットが低下します。また、MUSIC 文はワークエリア確保のために内部で CLEAR 文に相当することを行なっているので、HIMEM(CLEAR 文の第2パラメータに相当します）が807バイト小さく再設定され、変数はすべてクリアされます。

文 例

CALL MUSIC

デフォルトの設定をする。

CALL MUSIC(0,0,1,1,1,1,1,1,1,1,1)

1チャンネルずつ PLAY 文の文字列に割り当てる。

CALL PITCH

機 能

FM 音源の楽音の音高（ピッチ）を与えます。

書 式

CALL PITCH(<ピッチ1>[, <ピッチ2>])

解 説

FM 音源で発生する楽音の音高を指定します。<ピッチ>の範囲は410～459で単位は

[Hz]です。中央Cのすぐ上のA音(a2)の周波数で音高を表わします。トランスポーズとは独立に設定でき、デフォルトの値は440です。
ピッチ（またはトランスポーズ値）を変えるとリズム音や音程をもたない音を除くFM音の音の高さが変化します。PSG音源には作用しないので注意して下さい。
ピッチ2は指定しても無視されます。これはMSX-AUDIO拡張BASICとの互換性をとるために用意されているのものでMSX-MUSICでは意味を持ちません。
トランスポーズについてはTRANSPOSEの項を参照して下さい。

文　例

```
CALL PITCH(440)
```

PLAY

機　能

音楽をミュージックマクロランゲージ（MML）にしたがって演奏します。

書　式

PLAY[#<モード>,]<文字列1>[,<文字列2>[,<文字列3>]･･･[,<文字列13>]

解　説

PLAY文は音楽を演奏するもので、FM音源(9)、従来のPSG音源(3)の最大12声まで同時発声が可能です。<文字列>に書かれたミュージックマクロランゲージ（MML）にしたがって演奏します。
他の拡張命令と異なりCALL文は必要ありません。
<モード>は0から3までの値をとり、PLAY文の音源や動作モードを次のように設定します。

- 0や省略されたときはPSGのみが音源となり、文字列は最大3つまでとなります。従来のPLAY文と互換性があります。
- 1のとき、「Illegal function call」となります。これはMSX-AUDIO拡張BASICとの互換性をとるために用意されているのものでMSX-MUSICでは意味を持ちません。
- 2または3のとき、FM音源、リズム音、PSG音源を使用できます（2のときと3のときで動作に違いはありません)。

<文字列>と音源との関係は始めから順に、

<FM音源用文字列1>,･･･,<FM音源用文字列n>,

<リズム音用文字列>,
<PSG 音源用文字列 1>, <PSG 音源用文字列 2>,
<PSG 音源用文字列 3>

となります。n は MUSIC 文で設定されたミュージックマクロランゲージの個数です。MUSIC 文でリズム音を使用しないモードに設定した場合は、リズム音用文字列をカンマと共に省略しなければいけません。
例として初期設定の MUSIC 文に対する文字列の配列をあげると次のようになります。

<FM 音源用文字列 1>, <FM 音源用文字列 2>, <FM 音源用文字列 3>, <リズム音用文字列>, <PSG 音源用文字列 1>, <PSG 音源用文字列 2>, <PSG 音源用文字列 3>

文　例

```
PLAY # 2,"CD","EF","GA"
```

ミュージックマクロランゲージ（MML）の仕様

FM 音源、PSG 音源用 MML の仕様

表 7.23　FM 音源、PSG 音源用 MML の仕様一覧

文　字	意　味	値のとる範囲	初期設定値
Mn	エンベロープ周期の設定	1≦n≦65535	M255
Sn	エンベロープ形状の設定	0≦n≦15	S0
Vn	音量の設定	0≦n≦15	V8
Ln	長さの設定	1≦n≦64	L4
Qn	音の長さの割合	1≦n≦8	Q8
On	オクターブの設定	1≦n≦8	O4
>	オクターブを1つ上げる		
<	オクターブを1つ下げる		
Tn	テンポの設定	32≦n≦255	T120
Nn	n で指定された高さの音を発生する	0≦n≦96	
Rn	休符の設定	1≦n≦64	R4
A〜G	音程の発生		
+, #	音を半音上げる		
−	音を半音下げる		
.（ピリオド）	音符や休符の長さを 1.5 倍する		
= x;	パラメータ n を変数 x で設定する		
Xx;	文字変数 x に入っている MML を演奏する (*1)		
&	タイ、前後の音をつなぐ		

文　字	意　味	値のとる範囲	初期設定値
{}n	連符、n分音符を{}の中の音程の個数で等分にした音を発生する	1≦n≦64	Lnで設定された値
@n	n番の音色に切り換える	0≦n≦63	
@Vn	音量を細く設定する	0≦n≦127	
@Wn	nで指定された長さだけ状態を継続する	1≦n≦64	Lnで設定された値
Yr,d	音源LSIのレジスタrにdを書き込む		
Zd	「Illegal function call」になる (*2)		

*1 このマクロを指定した場合、このマクロ以降に何かマクロを書くことはできません。書いた場合はエラーとなります。

*2 MSX-AUDIOの場合、MIDIポートへの出力という意味ですが、MSX-MUSICの場合はエラーになります。

リズム音用MMLの仕様

リズム音の場合、1つのMMLで同時にいくつかの音を発生するため楽音用とは異なった記述様式をとります。まず鳴らしたい楽器を並べてその後に長さを記述します。

表7.24　リズム音用MMLの仕様一覧

文　字	意　味	値のとる範囲	初期設定値
B	バスドラム音を発生		
S	スネアドラム音を発生		
M	タムタム音を発生		
C	シンバル音を発生		
H	ハイハット音を発生		
!	直前の楽器の音量をアクセントボリュームにする		
n	直前までに書かれた楽音を発生し、n分音符分待つ	1≦n≦64	
Vn	アクセントの付いていない楽音の音量を設定する	0≦n≦15	8
@An	アクセントの付いている楽音の音量を設定する	0≦n≦15	

Tn、@Vn、Rn、=x;、Xx;、は FM音源用と同様です。

例

"BSH8H8S!H8H8"

- ■バス、スネア、ハイハットを鳴らし、8分音符分待ちます。
- ■ハイハットを鳴らし、8分音符分待ちます。
- ■スネアをアクセント付でハイハットと鳴らし8分音符分待ちます。
- ■ハイハットを鳴らし、8分音符分待ちます。

MML と各音源との対応

表7.25 MML と各音源との対応一覧

文　字	FM 音源	PSG 音源
Mn	*1	○
Sn	*1	○
Vn	○	○
Ln	○	○
Qn	○	*1
On	○	○
>	○	○
<	○	○
Tn	○	○
Nn	○	○
Rn	○	○
A～G	○	○
+,#	○	○
−	○	○
.	○	○
= x;	○	○
Xx;	○	○
&	○	○
{} n	○	*3
@n	○	*1
@Vn	○	*1
@Wn	○	*2
Yr,d	○	*1
Zd	*4	*1

*1 無視されます。

*2 Rn と同等です。

*3 PSG 音源に対しては使用できません。使用するとエラーになります。

*4 MSX-MUSIC ではエラーになります。

CALL PLAY

機　能

PLAY 文が音楽を演奏中かどうかを返します。

書　式

CALL PLAY(<PLAY 文のストリング番号>, <変数名>)

解　説

PLAY 文のミュージックキューの状態を調べ、各チャンネルが音楽を演奏中かどうかを判断し、演奏中であれば－1、そうでなければ0の値を返します。ただし、ストリング番号として0が与えられた場合はいずれかのストリングが演奏中であれば-1を、そうでなければ0を返します。

PLAY 文のストリング番号は、MUSIC 文で指定したストリング数＋3まで使えます。すなわち、MUSIC 文で指定した FM 音源に加え3チャンネルの PSG 音源について有効です。

文　例

```
CALL PLAY(0,A):PRINT A
```

CALL STOPM

機　能

バックグラウンドで実行中の PLAY 文の演奏を停止します。

書　式

CALL STOPM

解　説

バックグラウンドで実行中の PLAY 文の音楽の演奏を停止します。

CALL TEMPER

機　能

音律（テンペラメント）を与えます。

書　式

CALL TEMPER(<音律番号>)
音律番号　　　0～21

解　説

音律を与えるステートメントで、FM音源の楽音の音高に影響を与えます。
音律は1オクターブをどのような比率で12音に分割するかを決めるもので、古典音楽には古典音律が適していると言われます。
デフォルト値は9番の完全平均律です。

番　号	音　律
0	ピタゴラス
1	ミーントーン
2	ヴェルクマイスター
3	ヴェルクマイスター（*）
4	ヴェルクマイスター（別）
5	キルンベルガー
6	キルンベルガー（*）
7	ヴァロッティ・ヤング
8	ラモー
9	完全平均律（デフォルト）
10	純正律 c メジャー（a マイナー）
11	純正律 cis メジャー（b）
12	純正律 d メジャー（h）
13	純正律 es メジャー（c）
14	純正律 e メジャー（cis）
15	純正律 f メジャー（d）
16	純正律 fis メジャー（es）
17	純正律 g メジャー（e）
18	純正律 gis メジャー（f）
19	純正律 a メジャー（fis）
20	純正律 b メジャー（g）
21	純正律 h メジャー（gis）

（*）は平島達司氏による。

文　例

CALL TEMPER(0)

CALL TRANSPOSE

機 能

FM音源の楽音に対してセント単位で移調を与えます。

書 式

CALL TRANSPOSE(<トランスポーズ値1>[,<トランスポーズ値2>])

解 説

移調を行なうためのステートメントで、単位はセントです。これは、半音を100とした移調の単位で、1オクターブ上げるには、+1200を与えます。
トランスポーズ値として許される範囲は、±12799以内ですが、実際にはFM音源の音色によってある高さの範囲以外は制限されます。音高精度はLSIの制限により±2セント程度です。
トランスポーズはピッチとは独立して設定できます。MUSIC文による初期化の値は0です。ピッチについてはPITCH文を参照して下さい。
トランスポーズ値2は指定されても無視します。これはMSX-AUDIO拡張BASICとの互換性をとるために用意されているのものでMSX-MUSICでは意味を持ちません。

文 例

```
CALL TRANSPOSE(0)
CALL TRANSPOSE(0,700)
```

CALL VOICE

機 能

FM音源の各チャンネルに音色（ボイス）を直接に設定します。

書 式

CALL VOICE([<チャンネル1用のボイス>],[<チャンネル2用のボイス>],・・・,[<チャンネル9用のボイス>])

ボイス=@＋数式　または、
　　　配列変数名

解　説

音源 LSI の 9 チャンネルある FM 音源のそれぞれに音色を設定します。

音色の設定方法には 2 つあります。

システムに備えられている音色ライブラリを使う場合には、0〜63 の音色の番号を数式により指定します。この場合には数式の前に@記号をつけて次の配列変数名と区別します。

プログラムにより音色パラメータを与えて設定する場合には、配列変数に音色パラメータを入れてその配列変数名を指定します。

音色パラメータのフォーマットの詳細は VOICE COPY 文の解説を参照してください。

パラメータを省略したチャンネルの音色は変更されません。

音色ライブラリ（表 7.26 音色ライブラリ一覧表参照）のうち*印が付いていない音色や、配列変数で設定する場合は同時に 1 音色しか設定できません。複数設定しようとしたときはパラメータ列のいちばん右側のパラメータか最後に実行した CALL VOICE 文の設定のみが有効となります。

例

CALL VOICE(@26,@27)

チャンネル 1 の音色は 2 と同じ 27 番の音色になります。

CALL VOICE(@20,@10)

CALL VOICE(,, @21)

チャンネル 1 の音色は 3 と同じ 21 番となります。

CALL VOICE(A,,B)

チャンネル 1 の音色は 3 と同じ B という配列変数で設定される音色となります。

文　例

CALL VOICE(@0,@0,@0,,,,@7,@7,@7)

CALL VOICE(@0,@0)

略号は音色の名前としてライブラリに登録されているものです。

表7.26　音色ライブラリー覧表

音色番号	音色名	略　号
0	Piano 1*	Piano 1
1	Piano 2	Piano 2
2	Violin*	Violin
3	Flute 1*	Flute
4	Clarinet*	Clarinet
5	Oboe*	Oboe
6	Trumpet*	Trumpet
7	Pipe Organ 1	PipeOrgn
8	Xylophone	Xylophon
9	Organ*	Organ
10	Guitar*	guitar
11	Santool 1	Santool
12	Electric Pianol*	Elecpian
13	Clavicode 1	Clavicod
14	Harpsicode 1*	Harpsicd
15	Harpsicode 2	Harpscd2
16	Vibraphone*	Vibraphn
17	Koto 1	Koto
18	Taiko	Taiko
19	Engine 1	Engine
20	UFO	UFO
21	Synthesizer bell	SynBell
22	Chime	Chime
23	Synthesizer bass*	SynBass
24	Synthesizer*	Synthsiz
25	Synthesizer Percussion	SynPercu
26	Synthesizer Rhythm	SynRhyth
27	Harm Drum	HarmDrum
28	Cowbell	Cowbell
29	Close Hi-hat	ClseHiht
30	Snare Drum	SnareDrm
31	Bass Drum	BassDrum
32	Piano 3	Piano 3
33	Electric Piano 2*	Elecpia2
34	Santool 2	Santool2
35	Brass	Brass
36	Flute 2	Flute 2
37	Clavicode 2	Clavicd2
38	Clavicode 3	Clavicd3
39	Koto 2	Koto 2
40	Pipe Organ 2	PipeOrg2

音色番号	音色名	略　号
41	PohdsPLA	PohdsPLA
42	RohdsPRA	RohdsPRA
43	Orch L	Orch L
44	Orch R	Orch R
45	Synthesizer Violin	SynViol
46	Synthesizer Organ	SynOrgan
47	Synthesizer Brass	SynBrass
48	Tube*	Tube
49	Shamisen	Shamisen
50	Magical	Magical
51	Huwawa	Huwawa
52	Wander Flat	WnderFlt
53	Hardrock	Hardrock
54	Machine	Machine
55	Machine V	MachineV
56	Comic	Comic
57	SE-Comic	SE-Comic
58	SE-Laser	SE-Laser
59	SE-Noise	SE-Noise
60	SE-Star 1	SE-Star
61	SE-Star 2	SE-Star2
62	Engine 2	Engine 2
63	Silence	Silence

音色パラメータデータのフォーマット

音色パラメータは次の表のように１音色あたり32バイトのデータを使います。オペレータ０とオペレータ１のデータは同じものの繰り返しになっています。

表 7.27　音色パラメータデータのフォーマット

オフセット	内　容
	ヘッダ
0〜7	音色名
8〜9	ボイス移調
10	ビット0：アルゴリズム*
	ビット1〜3：フィードバック
	ビット4：固定ピッチ*
	ビット5：AMD/PMDロード可能*
	ビット6：PMD*
	ビット7：AMD*
11〜15	予約
	オペレータ0
16	bit0〜3：MULT
	but4：KSR
	bit5：EG
	bit6：PM
	bit7：AM
17	bit0〜5：トータルレベル
	bit6〜7：レベルキースケール
18	bit0〜3：ディケイレイト
	bit4〜7：アタックレイト
19	bit0〜3：リリースレイト
	bit4〜7：サスティンレベル
20	bit4〜7：ベロシティセンシビリティ（0〜8）*
21〜23	予約
	オペレータ1
24	bit0〜3：MULT
	bit4：KSR
	bit5：EG
	bit6：PM
	bit7：AM
25	bit0〜5：トータルレベル*
	bit6〜7：レベルキースケール
26	bit0〜3：ディケイレイト
	bit4〜7：アタックレイト
27	bit0〜3：リリースレイト
	bit4〜7：サスティンレベル
28	bit4〜7：ベロシティセンシビリティ（0〜8）*
29〜31	予約

*はMSX-AUDIOとの互換性のためにあるものです。MSX-MUSICでは意味を持ちません(無視されます)。

CALL VOICE COPY

機 能

音色パラメータデータの転送を行ないます。

書 式

CALL VOICE COPY(<パラメータ1>, <パラメータ2>)

パラメータ1＝@＋数式または配列変数名
パラメータ2＝@＋数式（結果は63でなければなりません）または配列変数名

解 説

配列と音色ライブラリ（0〜63）の間でのデータの転送を行ないます。パラメータ1の音色パラメータをパラメータ2に転送します。@と数式が指定されたときは、その数式の結果で指定される音色番号の音色データが対象となります。
ソース（パラメータ1）に指定できる音色番号は0〜63のうち*印の付いていない音色の番号ですが、ディスティネーション（パラメータ2）に指定できる音色番号は63に限ります。ソース(パラメータ1)に*印の付いている音色番号を指定すると「Illegal function call」となります
@記号がない場合の変数名は配列変数とみなされ、その内容が転送の対象になります。
1つの音色パラメータは32バイトの長さがあります。音色パラメータの詳細については、「表7.27 音色パラメータデータのフォーマット」を参照して下さい。

文 例

CALL VOICE COPY(@17,@63)
17番の音色データを63番に転送する。

DIM A%(16):CALL VOICE COPY(@28,A%)
28番の音色データを配列変数A%に転送する。

CALL VOICE COPY(A%,@63)
配列変数A%の音声データを63番に転送する。

3.2.5 MSX-AUDIO のステートメント

以下の MSX-AUDIO のステートメントは全て「Illegal function call」となります。
文法上の間違いがあっても同様です。

ADPCM / PCM 関係のステートメント

CONVA	CONVP	COPY PCM	LOAD PCM	PCM FREQ
PCM VOL	PLAY PCM	REC PCM	SAVE PCM	SET PCM

インスツルメント関係のステートメント

INMK	KEY ON / KEY OFF	MK PCM	MK TEMPO	MK VEL
MK VOICE	MK VOL			

MK 記録関係のステートメント

APPEND MK	CONT MK	MK STAT	PLAY MK	REC MK
RECMOD				

外部プログラムの呼び出し

SYNTHE	APEEK	APOKE

3.3 FM BIOS

3.3.1 概要

MSX-MUSICには、アプリケーションソフトウェア用のサービスルーチンとして、MSX-MUSIC FM BIOSが用意されています。多くの場合、MSXの周辺機器は拡張BIOSコールにより、そのデバイスをアクセスしますが、FM BIOSはその方法をとらず、FM BIOSが存在するスロットを捜し、特定の番地を直接コールする仕組みになっています。

高速処理を必要とする場合は、あらかじめスロットをイネーブルしておき、直接コールすることもできます。

商用のアプリケーションソフトウェアは必ずこのFM BIOSを使って、OPLLをアクセスして下さい。I/Oポートを直接操作した場合の互換性は保証できません。

この章ではFM BIOSの機能、コール方法などを解説します。FM BIOSコールを使用するにあたっては、OPLL (YM2413) の知識が必須ですから、「3.4 YM2413 (OPLL)」もあわせてご覧下さい。ただし、「3.4 YM2413 (OPLL)」はFM音源に関する解説ではないので、音の作り方などについては、専門の解説書をご参照下さい。

3.3.2 FM BIOSのサイズ

FM BIOSのサイズと格納番地は以下のとおりです。

番地	内容
4DFFH – 4C00H	音色データエリア (0.5Kバイト)
4C00H – 4100H	FM BIOS (2.75Kバイト)

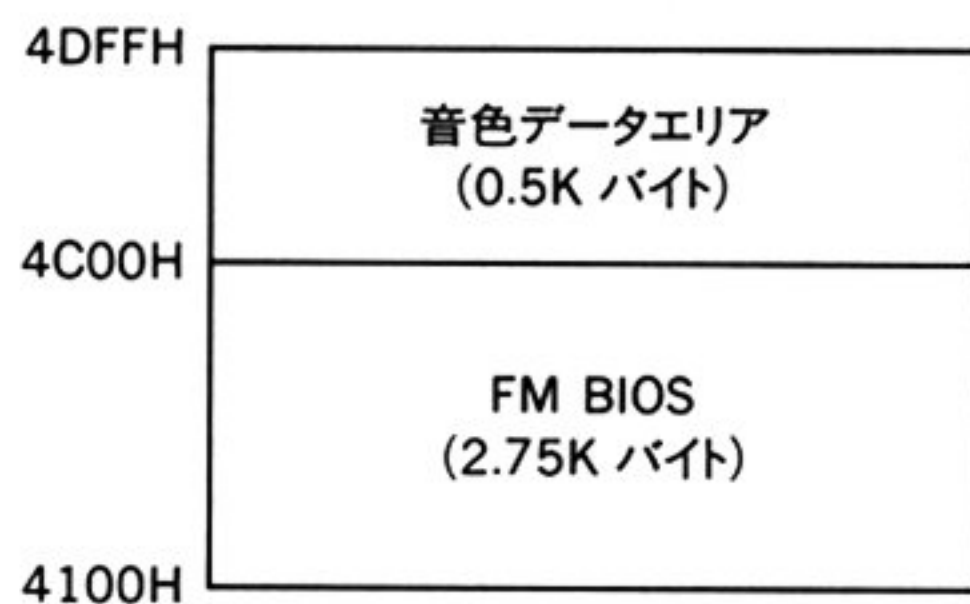



図7.19 FM BIOSのサイズ

FM BIOSはワークエリアとして160バイト、スタックエリアとして32バイト使用します。

3.3.3 FM BIOSの各機能

FM BIOSは、インタースロットコールにより呼び出されます。以下でエントリ名と、その入出力の設定レジスタ、内容が変化するレジスタ、およびその機能について解説します。

WRTOPL(4110H)

機能

OPLLレジスタへデータを書き込みます。

入力

A　　OPLLのレジスタ番号
E　　書き込むデータ

出力

なし

変更レジスタ

なし

解説

AレジスタでE指定した番号のOPLLレジスタに、Eレジスタの内容を書き込みます。このエントリルーチン内では、割り込み禁止・解除をしていないので、必要に応じてDI、EIを行う必要があります。

INIOPL(4113H)

機能

FM BIOSの環境を整えます。

入力

HL　　使用ワークエリアの先頭アドレス（偶数）

出 力

なし

変更レジスタ

AF、BC、DE、HL、IX、IY

解 説

HLレジスタで指定されたアドレス(偶数)に、FM BIOSで使用するワークエリアを設定し、全てのFM BIOS用のワークエリアおよびOPLLレジスタを初期化します。ワークエリアの先頭アドレス(偶数)は、【SLTWRK(0FD09H〜)】に格納されます。HLレジスタで指定されたエリアがRAMでない場合の動作保証はありません。

FM BIOSをコールするときは、最初にINIOPLを呼び出さなくてはなりません。このルーチンを1度も呼び出さずに、他のエントリルーチンを呼び出したときは、正常には動作しません。

このBIOSはEI状態でリターンします。

MSTART(4116H)

機 能

音楽の演奏を開始します。

入 力

HL	音楽データの先頭アドレス		
A	エンドレスフラグ		
	0	無限ループ	
	1〜254	繰り返し演奏回数の指定	
	255	設定してはならない	
(HL)*	0EH	リズムモード (FM6音+リズム部)	
	12H	メロディモード (FM9音)	

*(HL) はデータとして持っているので、新たに書き込む必要はありません。

出 力

なし

変更レジスタ

AF、BC、DE、HL、IX、IY

解　説

HLレジスタで指定されたアドレスに置かれている音楽データのヘッダ（後述）をもとに、MSX-MUSICのワークエリアを音楽演奏用に設定します。ただし、【H.TIMI(0FD9FH)】からOPLDRVを呼び出すように、あらかじめ設定しておかなければ、音楽の演奏はしません。
このBIOSはEI状態でリターンします。

MSTOP(4119H)

機　能

音楽演奏を中止します。

入　力

なし

出　力

なし

変更レジスタ

AF、BC、DE、HL、IX、IY

解　説

現在出力しているOPLLの全ての音の発生を止め、MSX-MUSICのワークエリアも初期化します。
このBIOSはEI状態でリターンします。

RDDATA(411CH)

機　能

ROM内の音色データを読み出します。

入　力

HL	データ読み出し用ワークエリアの先頭アドレス
A	音色ナンバー (0～63)

出　力

なし

変更レジスタ

F

解　説

ROMに内蔵されている音色を読み出し、指定のワークエリアに格納します。

OPLDRV(411FH)

機　能

OPLLドライバへのインタラプトのエントリアドレスです。

入　力

なし

出　力

なし

変更レジスタ

なし

解　説

音楽演奏を実際に行うOPLLドライバのエントリアドレスです。H.TIMIフックを書き変えて、このOPLDRVを呼び出すようにして下さい。ただし、このBIOSを使用する前に、必ず1度はINIOPLをコールして、FM BIOS用のワークエリアとOPLLレジスタを初期化して下さい。そうでなければ、その後の動作は保証できません。

TSTBGM(4122H)

機　能

演奏の終了をチェックします。

入　力

なし

出　力

A	0	演奏終了
	0以外	演奏中

変更レジスタ

AF

解　説

MSTARTにて演奏を開始した音楽が、現在演奏中かどうかを調べます。割り込み禁止・解除の設定はしていません。

3.3.4 FM BIOS で使用するデータ構造

FM BIOS で演奏可能なデータは、ヘッダ部とデータ部に分類されます。

1. ヘッダ部

このヘッダ部によって、リズム部を使用するかどうかの設定や各ボイスチャンネル用データの先頭アドレスを求めることができます。

■ FM6 音+リズム部構成の場合

データ先頭アドレス→	0EH	00H	リズム部データ先頭オフセット値
	2 バイト		FM1CH
	2 バイト		FM2CH
	2 バイト		FM3CH
	2 バイト		FM4CH
	2 バイト		FM5CH
	2 バイト		FM6CH
	データ部 格納エリア		

先頭オフセット値が 00H、00H なら、そのボイスは使用されません。

図 7.20　FM6 音＋リズム部構成のヘッダ

■ FM9 音構成の場合

データ先頭アドレス→	12H	00H	FM1CH データ先頭オフセット値
	2 バイト		FM2CH
	2 バイト		FM3CH
	2 バイト		FM4CH
	2 バイト		FM5CH
	2 バイト		FM6CH
	2 バイト		FM7CH
	2 バイト		FM8CH
	2 バイト		FM9CH
	データ部 格納エリア		

先頭オフセット値が 00H、00H なら、そのボイスは使用されません。

図 7.21　FM BIOS の FM9 音構成のヘッダ

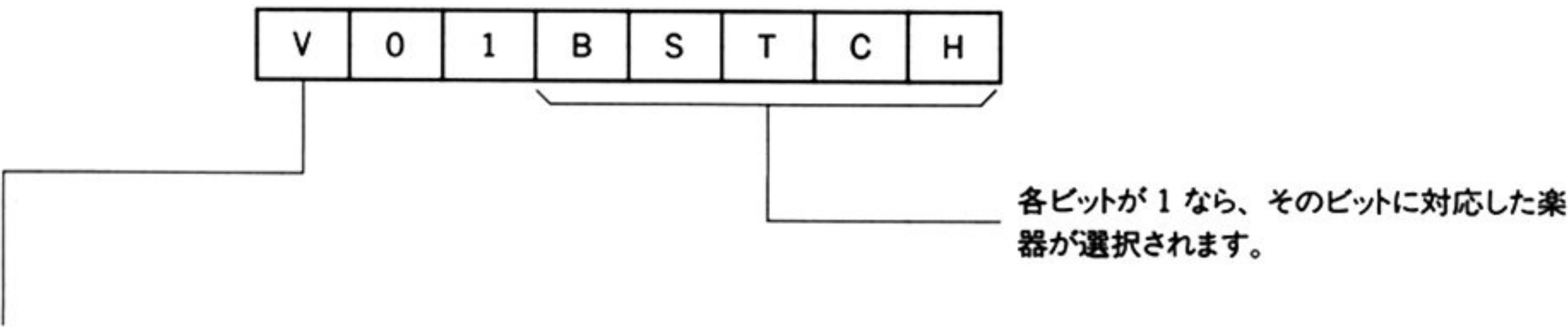
2. データ部

■メロディ部（FM 部）

アドレス	意　味
00H～5FH	音程を指定します。この数値自身が音域を含む全ての音階の状態を表します。続くデータが音長データです。音長データが 0FFH のときは、さらに次の 1 バイトも音長データとして加算されます。この音長読み出しは、その読み出された値が 255 以外になるまで続けられます。
60H～6FH	音量を指定します。この数値から 60H を引いた値が、実際の音量データです。
70H～7FH	音色を指定します。音色選択レジスタに書き込む実際の値は、この数値から 70H を引いた値です。
80H、81H	サスティンを指定します。80H でサスティン解除、81H でサスティン設定です。
82H	拡張音色を設定します。続く 1 バイトの値 (0～63) が、ROM の内蔵音色ナンバーです。この音色ナンバーの読み出しは最上位ビットが無視されます。
83H	ユーザ音色を指定します。続く 2 バイトの値 (下位、上位) が、音色データの格納されている先頭アドレスを示します。
84H	レガートオフです。音を音符毎に切ります。
85H	レガートオンです。音を切らずにつなぎます。
86H	Q 指定です。続く 1 バイト (1～8) で指定します。レガートオンのときは、Q 指定は実行しません。
87H～0FEH	未使用
0FFH	そのボイス毎のチャンネルデータの終了コードです。

■リズム部

V	0	1	B	S	T	C	H

各ビットが1なら、そのビットに対応した楽器が選択されます。

V=0　リズム発生を指定します。続くデータが音長データです。この音長データの読み出し方はメロディ部の音長の場合と同じです。

V=1　音量を指定します。続く1バイトが音量データ（0〜15）です。下位4ビットのみデータが有効です。

0FFH　リズム部のデータの終了コードです。

3. 音色データ格納形式

図7.22はユーザ音色データ（オリジナル音色データ）を作成し、このFM BIOSに組み込む場合のデータ形式です。ROMに内蔵されている音色データも、この形式で格納されています。

+0 / +1	AM	VIB	EG-TYP	KSR	MULTIPLE	
+2	KSL M		Total Level Modulator			
+3	KSL C		XX	DC	DM	Feedback
+4 / +5	Attack Rate			Decay Rate		
+6 / +7	Sustain level			Release Rate		

図7.22　FM BIOSの音色データ格納形式

音色データはOPLLのレジスタ00〜07の順番に並んでいます。1音のデータは8バイトで構成されます。したがって、ROM内蔵の音色データは全部で200H（512）バイトとなります。

3.3.5 FM BIOS 使用上の注意

1. ワークエリア

ワークエリアはいかなる場合においても、ページ1に指定することはできません。できるだけページ2かページ3に指定してください。ページ2とページ3にまたがって指定することも可能です。スロット管理を適切に行ったときに限り、ページ0にワークエリアを指定することができます。ただし、ページ0からページ1にまたがって指定はできません。

また、ワークエリアの先頭アドレスは必ず偶数でなければなりません。これはMSX-MUSICの拡張BASICと【SLTWRK(0FD09H)】を共用するための制限事項です。

3.3.6 FM BIOSの呼び出し方

全スロットをサーチして、401CH番地からに「OPLL」という文字列を持ったスロットを捜します。そして、「OPLL」という文字列を持ったスロットの4110H番地からがFM BIOSのエントリです。

1. 注意事項

FM BIOSのエントリINIOPLが呼ばれたとき、内部では他のスロットにFM BIOSが存在するかどうかを調べるためにENASLTを用いてスロットを切り換えて、スロット0から順次ROM内の401CH番地からのIDをチェックしています。もし、対象スロットが拡張されている場合には、同様に拡張スロットの0から走査しますが、この場合は走査終了時に拡張スロットセレクトレジスタを保存せず、常に拡張スロット3を選択したままにしてしまいます。

したがって、このチェックが終わった段階で、各スロットが拡張されていた場合には、それらスロットのページ1（4000Hから7FFFH）は拡張スロット3のままになっています。

FM BIOSはこのままの状態で呼び出し元のアプリケーションに戻るため、

■アプリケーションがページ2からFM BIOSを呼び出し、

■ページ1に現われているスロットが拡張スロットだった場合、

常にそのページは拡張スロット3が現われてしまいます。

2. 具体的な例

以下の2つの条件を同時に満たす場合、

■スロット0が拡張されてスロット0の拡張スロット0にBASICがあった場合

■スロット1からスロット3のどれかにFM BIOSがあった場合
BASICプログラムからUSR文で呼び出されるプログラムがFM BIOSのINIOPLをコールして帰ってきたとき動作異常が起こります。

この原因は、ページ1にはスロット0の拡張スロット3が現われているため、USR文で呼び出されたプログラムがBASICに戻るときに戻れないためです。

3. 対処方法

INIOPLを呼び出す前のページ1のスロットを覚えておき、INIOPLから戻ってきたときに元のスロットに戻します。

サンプルプログラム「OPLL.MAC」を参照して下さい。

3.4 YM2413(OPLL)

先に述べたとおり、MSX-MUSICではFM音源としてOPLL（YM2413）を採用しています。この章では、このFM音源LSIのレジスタなど、ソフトウェアを制作する上で必要なことを解説します。ただし、FM音源の理論や実際のプログラミングに関する解説ではないので、音の作り方などについては、専用の解説書をご参照下さい。

3.4.1 OPLLの概要

1. 概要

MSX-MUSICシステムとして採用されたYM2413 (OPLL) は、音源としてOPLL (YM2413) というYAMAHA独自のFM音源LSIを採用しています。YM2413は、DAコンバータや水晶発振回路を内蔵しているため従来の音源LSIに比べて、非常に容易にかつローコストで音源システムを組み立てることが可能です。さらにYM2413では、ソフトウェアの簡便さを図るため音色データをROMとして持ち、音色変更にともなうデータ変更を1度の音色選択操作ですませることができます。また、効果音や独自の音色も発音可能とするために1音色分の音色データレジスタも内蔵しています。

2. 特徴

- FM音源を採用し、リアルなサウンドを作ることが可能
- モード選択により9音同時発生あるいはメロディ音6音＋リズム音5音の2つのモードを選択可能（いずれの場合にも異音色可）
- 音色データ内蔵（メロディ音15音色、リズム音5音色）
- DAコンバータ内蔵
- 水晶発振回路内蔵
- ビブラート発振器／振幅変調発振器内蔵
- 入力TTLコンパチブル
- Si-gate NMOS LSI
- 5V単一電源

3. FM方式の概略

FM方式とは、Frequency Modulationすなわち周波数変調の意味で、変調によって生じる高調波を楽音の合成に利用したものです。この方式は比較的簡単な回路で、非調和音も含む高い高調波成分を持つ波形を発生させることができ、しかも変調指数と高調波のスペクトル分布の対応が非常に自然であるため、自然楽器の合成音から電子楽器まで、幅広い音作りが可能ということが確認されています。

FM方式は以下の式のように4つのパラメータで表現されます。

$$F = A\sin(\omega ct + I\sin\omega mt)$$

式7.1

ここでは、Aは出力振幅、Iは変調指数、またωc、ωmはそれぞれキャリア、モジュレータの各周波数です。この7.1式は次のように表現することもできます。

$$F = A[Jo(I)\sin\omega ct + J1(I)\{\sin(\omega c + \omega m)t - \sin(\omega c - \omega m)t\} + J2(I)\{\sin(\omega c + 2\omega m)t + \sin(\omega c - 2\omega m)t + ...]$$

式7.2

ここで、Jn(I)はn次の第1種Bessel関数です。7.2式からわかるように各倍音の振幅は、変調指数のBessel関数で表現されることになり、7.1式によるFM音源は特定の楽音や効果音の合成に非常に有効となることがわかります。ただし、これでは高調波が一様に分布しないためString系の音源には不向きとなります。そこで、考え出されたのが7.3式で表されるfeedback FMという方式です。

$$F = A\sin(\omega ct + \beta F)$$

式7.3

ここでβは帰還率です。このfeedback FMでは、高調波スペクトルが鋸歯状波となりString系の音作りも可能となります。

以上のような、FM方式を実現するためには、次の3つの機能ブロックが必要です。

1. ωtを発生させるphase generator（PG）
2. 振幅Aや変調指数Iを時間関数として得るためのenvelop generator（EG）
3. sinテーブル（sin）

以上の3つの構成要素を組み合わせて1つのユニットとして考えると、先のFM方式は図7.23のように表すことができます。したがって、このユニット（オペレータセル：OP）の考え方を用

いれば、FM方式の音作りは、ユニット内の周波数パラメータやEGパラメータの設定、そしてユニット間の組み合せのデータを作ればよいことになります。

a. $F(t)=A\ SIN\ (\omega ct+I\ SIN\ \omega mt)$　　b. $F(t)=A\ SIN(\omega t+\beta F(t))$

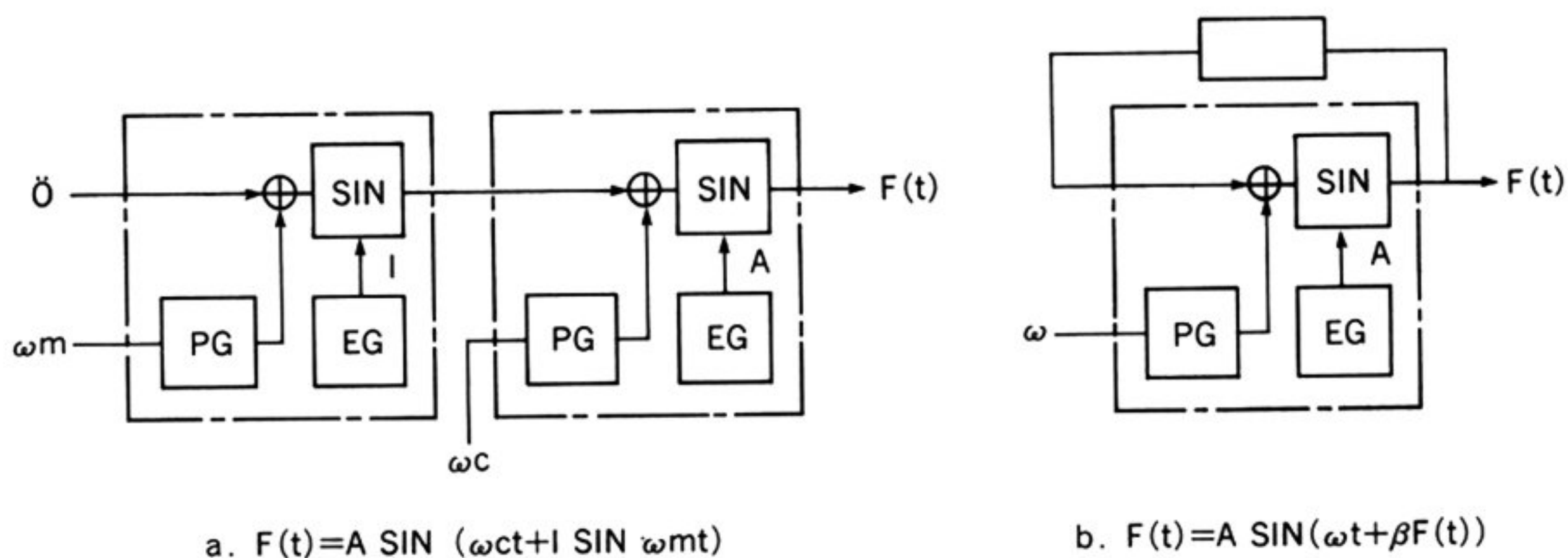



図7.23　ユニットセルによるFM方式の表現

3.4.2 機能概要

1. 主要機能

OPLLは9ビットのDAコンバータを内蔵したFM音源LSIであり、メロディ音を9音あるいはメロディ音6音+リズム音5音の2つの発音モードを持ち、両モードとも同時異音色発音が可能です。さらに、この両モードをソフトウェアで選択することも可能です。YM2413の特徴の1つは、音色ROMを内蔵していることです。この音色ROMは、表7.30のようにメロディに対して15音色、リズムに対して5音色用意されています。また、効果音や独自の音創りが可能なように1音色分の音色レジスタがあります。この音色レジスタの各パラメータは7.4式のE、ω1、I、ω2をコントロールすることにより、基本波ω1に対するいろいろな高調波を生成することができます。

$$FM = E\sin(\omega 1t+I\sin\omega 2t)$$

式7.4

OPLLは従来のFM音源と異なって、音色がROMとして内蔵されているため、プロセッサからの発音制御が大幅に簡素化されています。まず最初に、音色選択レジスタに希望の音色を登録します。その後Key-ON、F-Numberレジスタに所定の音程とタイミングでデータを書き込むことにより、発音を開始します。このとき、曲に合わせて適当にサスティンレジスタ、ボリューム

レジスタにデータを書き込めば、難なくプロセッサによる自動演奏を楽しむことができます。備え付けの音色以外の独自の音色を楽しむときには、先に述べた音色レジスタにデータをセットした後、音色選択レジスタを「0」にすることにより、オリジナルの音色を出すことができます。また、リズム音を発生したいときには、リズムコントロールレジスタの希望音源のビットをON/OFFすることにより、リズム音を付加することができます。この場合、Key-ON、F-Numberレジスタの7CH、8CH、9CH（アドレス$16、$17、$18、$26、$27、$28）は所定のデータを入力しておかねばなりません。

2. チャンネルとスロット

OPLLはFM音を9音（9チャンネル）発音することが可能で、1音あたり2オペレータセル持っています。ただし、オペレータセルはシステムで1つ持っているだけなので、FM9音の計算は、このオペレータセルをシリアルに18回通すことによってなされます。このオペレータセルを通す順番（スロット番号）は、レジスタ番号と対応しており、各音の発音コントロールはスロットと対応したレジスタを制御することになります。

また、F-Numberのようなチャンネルごとのデータは2つのスロットを制御します。この2つのスロット（第1、第2スロット）の関係は、FM変調モードにした場合は、第1スロットが必ず変調波に、そして第2スロットが搬送波になります。また、第1スロットはFeedback FMのモードにも設定できます。このモード設定については「KSL/TOTAL LEVEL/DISTORTION/FEED BACK LEVEL」の項を参照して下さい。

表7.28はチャンネルとスロットの関係を示します。

表7.28 チャンネルとスロット

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	スロット番号
1	2	3	1	2	3	4	5	6	4	5	6	7	8	9	7	8	9	チャンネル番号
1			2			1			2			1			2			チャンネルごとに見たときのスロット番号
20	21	22	20	21	22	23	24	25	23	24	25	26	27	28	26	27	28	チャンネルごとのデータとレジスタの関係（例 $20〜$28）

3. レジスタマップ

アドレス	D7	D6	D5	D4	D3	D2	D1	D0	
00 01	AM	VIB	EGTYP	KSR	MULTI				オリジナル 音色 レジスタ
02	KS Ⓜ		TLⓂ						
03	KS Ⓒ			DC	DM	FB			
04 05	AR				DR				
06 07	SL				RR				Ⓒ:Carrier Ⓜ:Modulator
OE			R	BD	SD			HH	リズム コントロール
OF	TEST								OPLLテスト データ （常時'0'）
10 18	F-Num. 0〜7								F-Number 下位 8 ビット
20 28			SUS ON/OFF	KEY ON/OFF	BLOCK 0〜2			F-Num 8	E-Number MSB、オクターブ指定 Key-On/Off レジスタ サンティン On/Off レジスタ
30 38	INST				VOL				音色セレクト&ボリューム レジスタ

図7.24 OPLL のレジスタマップ

36		BD-VOL	リズム音ボリュームレジスタ
37	HH-VOL	DS-VOL	
38	TOM-VOL	T-CY-VOL	

図7.25 リズムモード時のレジスタマップ（Addr =$0E、D5 =「1」のとき）

表7.29 OPLLレジスタの内容

	アドレス	ビット	
1	00、01	D_7	振幅変調のON/OFF
		D_6	ビブラートのON/OFF
		D_5	持続音と減衰音の切り換え(0=減衰音、1=持続音)
		D_4	RATEのキースケール
		D_0～D_3	搬送波と変調波の周波数の制御
2	02、03	D_6、D_7	LEVELのキースケール
3	02	D_0～D_5	変調波のトータルレベル。変調指数の制御
4	03	D_3、D_4	搬送波、変調波の歪波形（半波整流）のON/OFF
		D_0～D_2	Feed back FMの帰還係数
5	04、05	D_4～D_7	アタック時のエンベロープの変化割合制御
		D_0～D_3	ディケイ時のエンベロープの変化割合制御
6	06、07	D_4～D_7	ディケイからサスティンへ移るレベルの指示
		D_0～D_3	リリース時のエンベロープの変化割合制御
7	0E	D_5	リズム音のモード選択(0=メロディモード、1=リズム音モード)
		D_0～D_4	各リズム楽器のON/OFF
8	10～18	D_0～D_7	F-Number下位8ビット
9	20～28	D_5	サスティンのON/OFF
		D_4	Key ON/OFF
		D_1～D_3	オクターブ指定
		D_0	F-Number MSB
10	30～38	D_4～D_7	音色ナンバー（INST.）
		D_0～D_3	ボリュームデータ

表7.30 音色データ

INST	音　色	INST	音　色
0	オリジナル音色	8	オルガン
1	バイオリン	9	ホルン
2	ギター	A	シンセ
3	ピアノ	B	ハープシコード
4	フルート	C	ビブラフォン
5	クラリネット	D	シンセベース
6	オーボエ	E	ウッドベース
7	トランペット	F	エレキベース

	音　色
DB	バスドラム
SD	スネアドラム
TOM	タム
T-CY	トップシンバル
HH	ハイハット

3.4.3. 動作説明

OPLLの全機種は、プロセッサからレジスタアレーへのデータの書き込むことによって制御されます。この書き込まれたデータによって、楽音のエンベロープの形状や変調度、周波数および発音モードなどが決定されます。そして、このデータの組み合せが、ピアノやバイオリンなどの音を発生させることになります。しかし、その組み合せには非常に多く、複雑であるため、OPLLでは音色レジスタに音色ナンバー（INST）をセットすることで音を発生することができます。

1. レジスタ

レジスタは図7.24のアドレスマップで与えられる計271ビットのエリアを持っています。ここでいうアドレスとはOPLL内で各レジスタに割り当てられたサブアドレスであり、楽音データはこのサブアドレスを通してレジスタ内に書き込まれることになります。

したがって、あるデータをOPLLに格納したい場合は、まず、そのデータをしまうサブアドレスデータを送り、次に楽音データを送ります。ただし、同一サブアドレスを何度もアクセスする場合は、最初にサブアドレスデータを送るだけで、以後はアドレスデータを送ることなしに、楽音データを送って、データを更新することができます。

なお、全レジスタとも初期設定のときには「0」にセットされます。

AM/VIB/FG-TYP/KSR/MULTIPLE　$00 $01

このレジスタでは、エンベロープの形状やF-Numberで与えられる周波数データを楽音の周波数成分に見合った搬送波（$01）、変調波（$00）の周波数に変換するための倍率を制御します。

$00、$01

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
AM	VIB	EG-TYP	KSR	MULTI			
				2^3	2^2	2^1	2^0

D_0～D_3
(MULTIPLE)　表7.31で与えられる倍率によって搬送波、変調波の周波数を制御します。

例

$F(t) = E\sin(\omega f t + I\sin(7\omega f t))$

F-Number による周波数	ωf
搬送波の MULTIPLE	1
変調波の MULTIPLE	7

表 7.31 搬送波、変調波の周波数を求めるための倍率

MUL	倍率	MUL	倍率	MUL	倍率	MUL	倍率
0	½	4	4	8	8	C	12
1	1	5	5	9	9	D	12
2	2	6	6	A	10	E	15
3	3	7	7	B	10	F	15

D_4 (KSR)　RATE のキースケールを与えます。自然楽器では、おおむね音程が高くなるにしたがって、音の立ち上がり、立ち下がりは速くなります。この現象をシミュレートするのが RATE のキースケールであり、表 7.32 の値が各々の音程に対してスピードのオフセットとして加えられます。したがって、実際の RATE は ADSR に対して設定した RATE にこのオフセットを加えたものになります。

$RATE = 4 \times R + Rks$　　R は ADSR での設定値
Rks はキースケール　オフセット値
ただし、$R = 0$ のときは $RATE = 0$

表 7.32 キースケールオフセット

0		1		2		3		4		5		6		7		オクターブ
0		1		2		3		4		5		6		7		BLOCK データ
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	F-Num・MSB
0	0	0	0	1	1	1	1	2	2	2	2	3	3	3	3	$D_4 = 0$ キースケールオフセット
0	1	2	3	4	5	6	7	8	9	12	11	12	13	14	15	$D_4 = 1$ (Rks)

D_5 (EG-TYP)　持続音か減衰音かの切り換えをします。

$D_5 =$「0」のとき、減衰音

$D_5 =$「0」のとき、持続音

この発音モードの違いは、RELEASE RATE の使用法が異なっているためで、その様子を図 7.26 に示します。

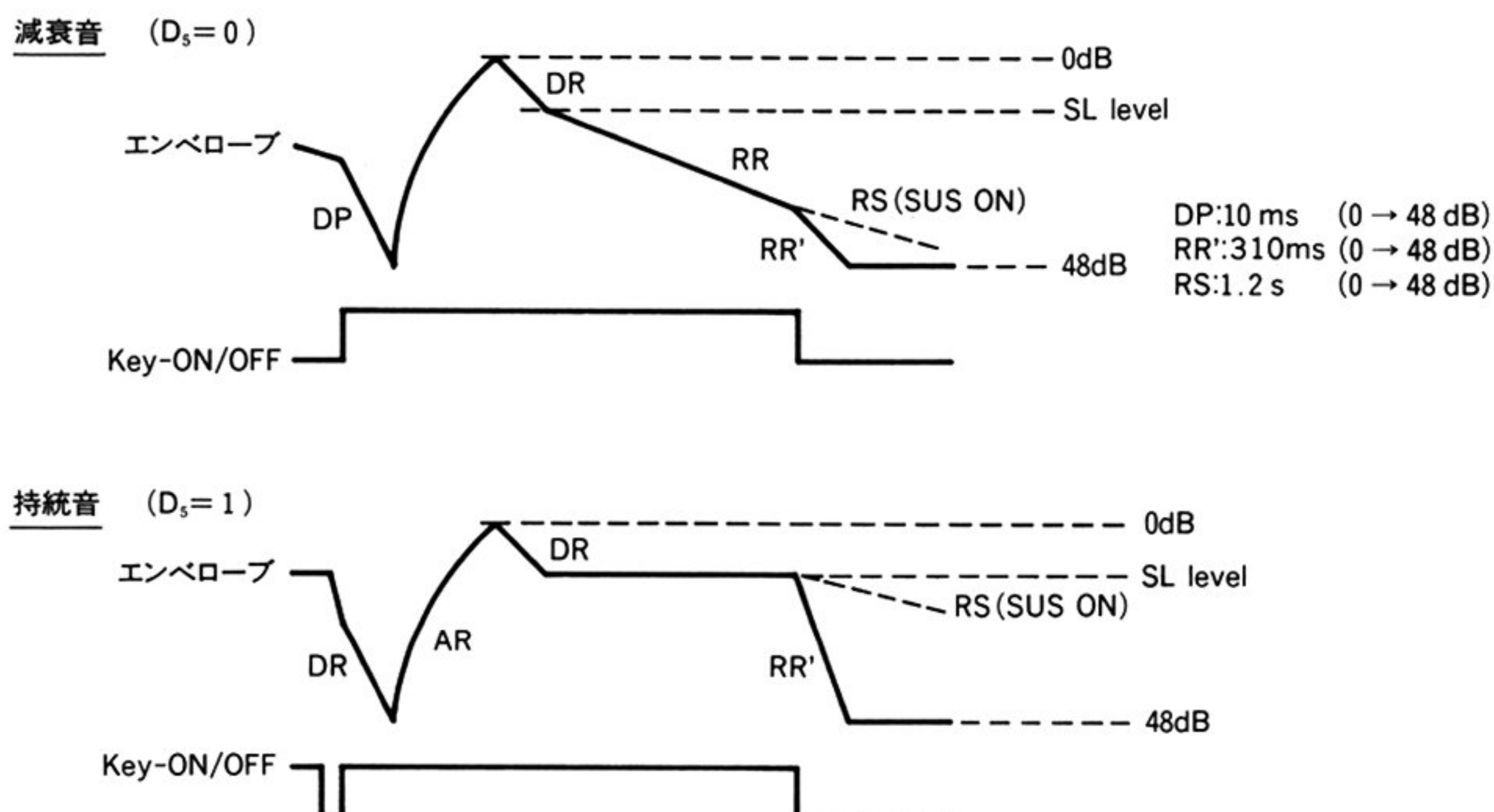



図 7.26 OPLL の減衰音、持続音のエンベロープ

D_6 (VIB) ビブラートの ON / OFF スイッチです。このビットを「1」にすると、そのスロットにはビブラートがかかります。このときの周波数は 6.4Hz (@ϕM = 3.6MHz) です。

D_7 (AM) 振幅変調の ON / OFF スイッチです。このビットが「1」にセットされたときには、そのスロットには振幅変調がかかります。振幅変調の周波数は 3.7Hz (@ϕM = 3.6MHz) です。

KSL/TOTAL LEVEL/DISTORTION/FEED BACK LEVEL $02 $03

トータルレベルは、エンベロープジェネレータの出力に対して減衰量を加算し、変調度(音色)の制御をするために用いられます。また、レベルキースケール (KSL) は RATE のキースケール同様、自然楽器では音程が上がるにつれて、出力レベルは低下する傾向にあることをシミュレートするものです。

<table>
<tr><td></td><td>D_7</td><td>D_6</td><td>D_5</td><td>D_4</td><td>D_3</td><td>D_2</td><td>D_1</td><td>D_0</td></tr>
<tr><td>$02</td><td colspan="2" rowspan="2">KSL</td><td colspan="6">Total Level Ⓜ</td></tr>
<tr><td>$03</td><td></td><td>DC</td><td>DM</td><td colspan="3">FB</td></tr>
</table>

$02

D_6～D_5（Total Level）

最小分解能は0.75dBで、最大47.25dBまで変調度を絞り込むことができます。

表7.33　トータルレベル

	D_5	D_4	D_3	D_2	D_1	D_0	
減衰量	24	12	6	3	1.5	0.75	（単位：dB）

D_6、D_7（KSL）　キースケールを制御するビットです。キースケールのモードは、音程が上がるほどレベルは減衰し、その減衰量は、1.5dB/OCT、3dB/OCT、6dB/OCTおよび減衰無しの4種類です。

表7.34　減衰量

D_7	D_6	減衰量
0	0	0
1	0	1.5dB/OCT
0	1	3dB/OCT
1	1	6dB/OCT

表 7.35　3dB/OCT の場合の各 F-Number での減衰量

F-Number OCT	0 8	1 9	2 10	3 11	4 12	5 13	6 14	7 15
0	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000	0.000 0.000
1	0.000 0.000	0.000 0.075	0.000 1.125	0.000 1.500	0.000 1.875	0.000 2.250	0.000 2.625	0.000 3.000
2	0.000 3.000	0.000 3.750	0.000 4.125	0.000 4.500	0.000 4.875	1.125 5.250	1.875 5.625	2.625 6.000
3	0.000 6.000	0.000 6.750	0.000 7.125	1.875 7.500	3.000 7.875	4.125 8.250	4.875 8.625	5.625 9.000
4	0.000 9.000	0.000 9.750	3.000 10.125	4.875 10.500	6.000 10.875	7.125 11.250	7.875 11.625	8.625 12.000
5	0.000 12.000	3.000 12.750	6.000 13.125	7.875 13.500	9.000 13.875	10.125 14.250	10.875 14.625	11.625 15.000
6	0.000 15.000	6.000 15.750	9.000 16.125	10.875 16.500	12.000 16.875	13.125 17.250	13.875 17.625	14.625 18.000
7	0.000 18.000	9.000 18.750	12.000 19.125	13.875 19.500	15.000 19.875	16.125 20.250	16.875 20.625	17.625 21.000

単位（dB）

注　意　F-Number は上位 4 ビットの値
1.5dB/OCT は上記の 1/2 倍
6dB/OCT は上記の 2 倍

$03

D_3　DM　変調波を半波整流する。

D_4　DC　運送波を半波整流する。

D_0〜D_2（FEEDBACK）　第 1 スロットのフィードバック FM 変調の変調度を与えます。

表 7.36　変調度

	0	1	2	3	4	5	6	7
変調度	0	$\pi/16$	$\pi/8$	$\pi/4$	$\pi/2$	π	2π	4π

ATTACK/DECAY REAT　$04　$05

アタックレイトは音の立ち上がり時間の設定をします。また、ディケイレイトは、アタック後の減衰時間を決めます。各 RATE の時間設定は表 7.37 のとおりです。

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
	AR				DR				
$04	2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0	変調波
$05	2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0	搬送波

SUSTAIN LEVEL/RELEASE RATE　$06 $07

サスティンレベルは、持続音の場合は、ディケイモードでの減衰がこのレベルに到達するとその後はそのレベルを保持するという変化点を指し、減衰音の場合は、ディケイモードからリリースモードへの変化点を与えます。

リリースレベルは、持続音の場合はKeyをOFFしたときに、音が消えてゆく様子を定義するレイトであり、減衰音のときはサスティンレベルの前の減衰をディケイレイトで表し、サスティンレベル後の減衰をこのリリースレイトで表します。

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
	SL				RR				
$06	24dB	12dB	6dB	3dB	2^3	2^2	2^1	2^0	変調波
$07	24dB	12dB	6dB	3dB	2^3	2^2	2^1	2^0	搬送波

注　意　リリースレイトの減衰時間は、ディケイレイトの表と同じ。

下記のレイトは、キースケール後のレイトです。また、レイトの値を上位4ビット(RM)と下位2ビット (RL) に分割してRM-RLと表しています。RATE＝RM×4＋RL

表7.37 各レイトでの立ち上がり、立ち下がり時間

EG DECAY TIME RATE RM-RL		[mS] 0dB-48dB	[mS] 10%-90%	EG ATTACK TIME RATE RM-RL		[mS] 0dB-48dB	[mS] 10%-90%
15	3	1.27	0.52	15	0	0.00	0.00
15	2	1.27	0.52	15	0	0.00	0.00
15	1	1.27	0.52	15	0	0.00	0.00
15	0	1.27	0.52	15	0	0.00	0.00
14	3	1.47	0.60	14	3	0.14	0.10
14	2	1.71	0.68	14	2	0.18	0.12
14	1	2.05	0.82	14	1	0.22	0.14
14	0	2.55	1.03	14	0	0.28	0.18
13	3	2.94	1.21	13	3	0.30	0.22
13	2	3.42	1.37	13	2	0.34	0.22
13	1	4.10	1.65	13	1	0.42	0.26
13	0	5.11	2.05	13	0	0.50	0.32

EG DECAY TIME RATE RM-RL		[mS] 0dB-48dB	[mS] 10%-90%	EG ATTACK TIME RATE RM-RL		[mS] 0dB-48dB	[mS] 10%-90%
12	3	5.87	2.41	12	3	0.54	0.36
12	2	6.84	2.74	12	2	0.60	0.38
12	1	8.21	3.30	12	1	0.70	0.44
12	0	10.22	4.10	12	0	0.84	0.54
11	3	11.75	4.83	11	3	0.97	0.64
11	2	13.68	5.47	11	2	1.13	0.72
11	1	16.41	6.60	11	1	1.37	0.84
11	0	20.44	8.21	11	0	1.69	1.09
10	3	23.49	9.65	10	3	1.93	1.29
10	2	27.36	10.94	10	2	2.25	1.45
10	1	32.83	13.19	10	1	2.74	1.69
10	0	40.87	16.41	10	0	3.38	2.17
9	3	46.99	19.31	9	3	3.86	2.57
9	2	54.71	21.88	9	2	4.51	2.90
9	1	65.65	26.39	9	1	5.47	3.38
9	0	81.74	32.83	9	0	6.76	4.34
8	3	93.97	38.62	8	3	7.72	5.15
8	2	109.42	43.77	8	2	9.01	5.79
8	1	131.31	52.78	8	1	10.94	6.76
8	0	163.49	65.65	8	0	13.52	8.96
7	3	187.95	77.24	7	3	15.45	10.30
7	2	218.84	87.54	7	2	18.02	11.59
7	1	262.61	105.56	7	1	21.88	13.52
7	0	326.98	131.31	7	0	27.03	17.38
6	3	375.90	154.48	6	3	30.90	20.60
6	2	437.69	175.07	6	2	36.04	23.17
6	1	525.22	211.12	6	1	43.77	27.03
6	0	653.95	262.61	6	0	54.07	34.76
5	3	751.79	308.96	5	3	61.79	41.19
5	2	875.37	350.15	5	2	72.09	46.34
5	1	1050.45	422.24	5	1	87.54	54.07
5	0	1307.91	525.22	5	0	108.13	69.51
4	3	1503.58	617.91	4	3	123.58	82.39
4	2	1750.75	700.30	4	2	144.18	92.69
4	1	2100.89	844.48	4	1	175.07	108.13
4	0	2615.82	1050.45	4	0	216.27	139.03
3	3	3007.16	1235.82	3	3	247.16	164.78
3	2	3501.49	1400.60	3	2	288.36	185.37
3	1	4201.79	1688.95	3	1	350.15	216.27
3	0	5231.64	2100.89	3	0	432.54	278.06
2	3	6014.32	2471.64	2	3	494.33	329.55
2	2	7002.98	2801.19	2	2	576.72	370.75
2	1	8403.58	3377.91	2	1	700.30	432.54
2	0	10463.30	4201.79	2	0	865.08	556.12
1	3	12028.60	4943.28	1	3	988.66	659.11
1	2	14006.00	5602.39	1	2	1153.43	741.49
1	1	16807.20	6755.82	1	1	1400.60	865.08
1	0	20926.60	8403.58	1	0	1730.15	1112.24

注　意　レイトが「0」の場合は、エンベロープは変化しません。

RHYTHM　$0E

リズムのモード選択と各リズム楽器のON/OFFをコントロールします。

$0E	D_5	D_4	D_3	D_2	D_1	D_0
	RHYTHM	BD	SD	TOM	TOP-CY	HH

D_0〜D_5（RHYTHM）　D5 =「1」のとき、OPLLはリズム音モードになり、7〜9チャンネル（「3.4.2　2. チャンネルとスロット」参照）はリズム音のチャンネルとなります。したがって、楽音（メロディ部）は6音に制限されます。D_0〜D_1は各リズム楽器のON/OFFを制御します。このため$26、$27、$28のKEY ONビットは常に「0」にしておく必要があります。また、13〜18の各スロットはリズム音と表7.38のような対応をしており、F-Numberのデータは各リズム音にマッチした値を入力しなければなりません。

表7.38　リズムスロットと周波数データ

楽器	スロット
DB	13、16
SD	17
TOM	15
TOP-CYM	18
HH	14

アドレス	データ
$16	$20
$17	$50
$18	$C0
$26	$05
$27	$05
$28	$01

TEST　$0F

このアドレスはLSIの内部動作をテストするときに使用しており、「0」以外では正常動作しません。

BLOCK/F-Number/SUS/KEY　$10～$28

音程、音階を決めるデータです。F-Number は$10～$18 のレジスタと$20～$28 のレジスタにまたがっています。

$10～$18

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
F-Number							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$20～$28

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
		SUS ON/OFF	KEY ON/OFF	BLOCK			F-NUMBER
				2^2	2^1	2^0	2^8

$10～$18

D_0～D_7 ($10～$18)、D_0 ($20～$28) (F-Number)
$10～$18 の 8 ビットと$20～$28 の下位 1 ビットの計 9 ビットで F-Number を表します。この F-Number は音階を与えるデータで、後述する方法でその値を決めます。

D_1～D_3 (BLOCK)　オクターブ情報を与えます。

$20～$28

D_4 (KEY ON)　鍵盤の ON / OFF に相当するビットです。このビットを「1」にすると、そのチャンネルが ON となり、発音します。「0」で KEY OFF です。

D_5 (SUS ON / OFF)　このビットを「1」にすると KEY OFF 時の RR が 5 になります。

■ F-Number / BLOCK

OPLL では、必要な周波数はその周波数に応じた位相の増分を与えることにより、得ることが

できます。そして、この増分はF-NumberとBLOCKおよびMULTIPLE情報によって決められます。

そこで、まず希望周波数の増分を求めます。これは次式で与えられます。

$\Delta P = fmus \times 2^{18} / fsam$

$fsam = fM / 72$

fmus　希望周波数

fsam　サンプリング周波数（50KHz）

fM　入力クロック周波数（3.6MHz）

式7.5 希望周波数の増分

これで位相の増分は求めれらますが、この値を管理するのはビット数が多くて大変なため、増分は1オクターブ分のデータのみとし、各オクターブに対してはその増分をシフト(2倍、4倍･･･)することによって求めます。これにより増分は次のように表現できます。

$\Delta P = 2B \times F' \times MUL$

B　オクターブ情報

F'　1オクターブ内に制限した増分

MUL　MULTIPLEデータ

式7.6 シフトによる位相の増分

7.5式、7.6式と増分（F'）を9ビットで表すということから、F-NumberとBLOCKは次のように表現されます。

$F = (fmus \times 218 / fsam) / 2b\text{-}1$　　(@MUL = 1)

F　F-Numberデータ

b　BLOCKデータ

式7.7 F-NumberとBLOCK

表7.39 F-Number（その1）

音階	周波数 (4oct)	F-Number	$20〜$28	$10〜$18							
			D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
C#	277.2	181	0	1	0	1	1	0	1	0	1
D	293.7	192	0	1	1	0	0	0	0	0	0
D#	311.1	204	0	1	1	0	0	1	1	0	0
E	329.6	216	0	1	1	0	1	1	0	0	0
F	349.2	229	0	1	1	1	0	0	1	0	1
F#	370.0	242	0	1	1	1	1	0	0	1	0
G	392.0	257	1	0	0	0	0	0	0	0	1
G#	415.3	272	1	0	0	0	1	0	0	0	0
A	440.0	288	1	0	0	1	0	0	0	0	0
A#	466.2	305	1	0	0	1	1	0	0	0	1
B	493.9	323	1	0	1	0	0	0	0	1	1
C	523.3	343	1	0	1	0	1	0	1	1	1

表7.40 F-Number（その2）

音階	周波数 (4oct)	F-Number	$20〜$28	$10〜$18							
			D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
G	392.2	257	1	0	0	0	0	0	0	0	1
G#	415.3	272	1	0	0	0	1	0	0	0	0
A	440.0	288	1	0	0	1	0	0	0	0	0
A#	466.2	305	1	0	0	1	1	0	0	0	1
B	493.9	323	1	0	1	0	0	0	0	1	1
C	523.3	343	1	0	1	0	1	0	1	1	1
C#	554.4	363	1	0	1	1	0	1	0	1	1
D	587.3	385	1	1	0	0	0	0	0	0	1
D#	622.2	408	1	1	0	0	1	1	0	0	0
E	659.3	432	1	1	0	1	1	0	0	0	0
F	698.5	458	1	1	1	0	0	1	0	1	0
F#	740.0	485	1	1	1	1	0	0	1	0	1

INSTRUMENT/VOLUME　$30〜$38

音色（ROM 15音色、オリジナル音色）、音量を決めるデータを設定します。

$30～$38	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	INST				VOL			

D_4～D_7（INST）　この4ビットで以下の音色が決定されます。

表7.41　INSTの値と音色

INST	音　色	INST	音　色
0	オリジナル音色	8	オルガン
1	バイオリン	9	ホルン
2	ギター	A	シンセ
3	ピアノ	B	ハープシコード
4	フルート	C	ビブラフォン
5	クラリネット	D	シンセベース
6	オーボエ	E	ウッドベース
7	トランペット	F	エレキベース

D_0～D_3（VOL）　各音色の音量を決めます。最少分解能は3dBで、最大45dBまで音量を絞り込めます。

D_3	D_2	D_1	D_0
24dB	12dB	6dB	3dB

リズムモード（addr =$OE、Ds = 「H」）時は、$36～$38は、次のように各リズムのボリュームを設定できます。

	D_7 D_6 D_5 D_4	D_3 D_2 D_1 D_0
$36	――――	DB
$37	HH	SD
$38	TOM	T-CYM

2. フェイズジェネレータ (PG)

フェイズジェネレータは必要な周波数に応じた増分を単位時間ごとにアキュムレートして位相値を得る回路です。この増分はレジスタから送られてくる周波数情報 (F-Number、BLOCK、MULTIPLE)から作られます。さらに、ビブラート発振器を内蔵しているため、この発振器の出力と周波数情報とを組み合わせることにより、ビブラート効果を作り出します。

3. エンベロープジェネレータ (EG)

エンベロープジェネレータ (以下、EG) は、ATTACK、DECAY、RELEASE の各 RATE、Sustain Level、Total Level などでコントロールされ、音色、音量の経時変化を与えます。そして、そのダイナミックレンジは 48dB (分解能 0.325dB) あります。EG は対数表示であり、また減衰量で表されます。その一般的な波形は図 7.27 のとおりです。この波形で特徴的なのは、アタック時は指数関数的に変化し、それ以外では直線的に変化する点です。また、アタックからディケイへの切り換えは、0dB に達したときに起こり、ディケイからサスティンへは、サスティンレベルに到達したときに起こります。そして、リリースへの移行は Key が OFF されたときに起こります。トータルレベル、レベルキースケール、振幅変調などの効果は、その設定値を EG に加えることによりエンベロープの波形を変化させます。

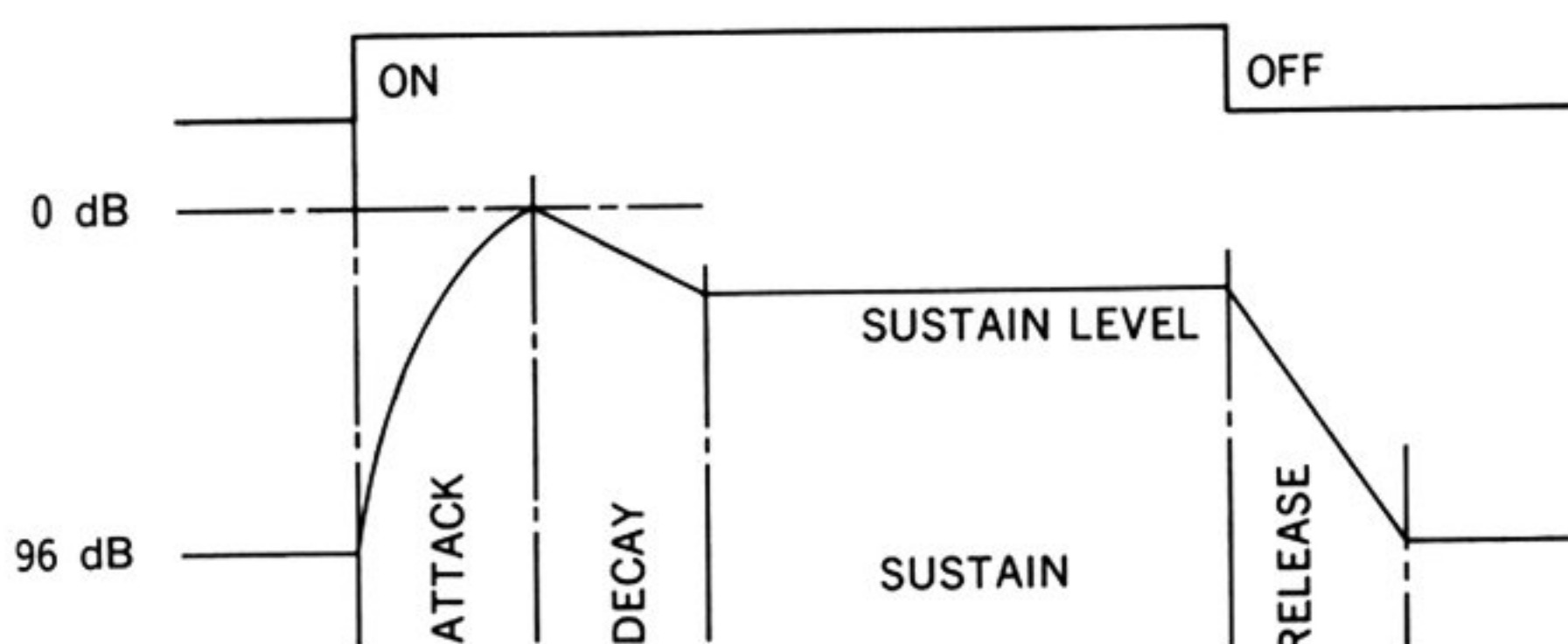



図 7.27　OPLL のエンベロープ波形

3.4.4 OPLL での楽音の作り方

この章では、OPLL のオリジナル音色レジスタに、どのような DATA を入力すると、ピアノやブラスなどの楽音を作ることができるかを説明します。

1. 音作りの考え方

FM 方式での音作りの基本は、まず作りたい楽器の特徴をよく理解することです。例えば、ピアノであれば、鍵盤を押したときに、鋭い音の立ち上がりがあり、その後、押鍵を続けていれば、徐々に音が消えて行くエンベロープを持っています。また、倍音の構成も立ち上がり時に多く、時が経つに連れて倍音の数は少なくなり、一定の倍音構成に近づいて行きます。

以上のような特徴をつかんだ後、FM の式で以下にして実現するかを考えます。エンベロープの特徴から出力振幅を、そして倍音構成から変調指数を決めることができます。また、倍音の構成はオペレータの周波数も関与していますから、周波数比もある程度決めることができます。このように、各楽音の特徴から FM の各パラメータをおおまかに決め、その次に音を聞きながら細部をつめてゆくようにすれば、望みどおりの音色を得ることができます。

2. 音作りの基本

FM 音源とは、モジュレータによってキャリアを変調することから生じる効果を利用したものです。したがって、FM の基本式パラメータ(キャリアの出力レベル、モジュレータの出力レベルモジュレータのフィードバックレベル、キャリアの周波数、モジュレータの周波数) を上手に扱うことにより、各楽音のピッチ、音色、音量のすべてを決めることができます。この FM の各パラメータと OPLL のパラメータとの関係は、表 7.42 のとおりです。

3. 音作りの例

表 7.42 OPLL の音作りの基本

項　目	関与するパラメータ	MIN ←(音の変化)→ MAX
キャリアの出力レベル	TOTAL LEVEL	音　量　小←→音　量　大
モジュレータの出力レベル	(A、D、S、R の各データ) (Key Scale データ)	丸い音色←→明るい音色
モジュレータのフィードバックレベル	FB	普通の音色←→鋭い音色 (Noise)
キャリアの周波数	MULTIPLE	ピッチ低←→ピッチ高
モジュレータの周波数	(BLOCK / F-Number)	近い倍音←→離れた倍音

1. エレクトリックピアノ

1. オペレータの周波数の決定

整数倍の高調波をすべて出すために、2つのオペレータともにMULTIPLEは「1」を使います。

2. オペレータの出力レベル

今度はモジュレータの出力を変更して音色を調整します。このとき、オペレータ1のレベルを決めるときには、低音部がまずピアノらしいリッチな高調波を得られるように設定し、それから高音にかけての変化はオペレータ1のレベルスケーリングで調整します。高音部ではほとんどSIN波になる位までレベルスケーリングをする必要があります。

3. EGの設定

ここでは音量と音色のエンベロープを決めます。まず、オペレータ2はアタックを鋭く、しかもある程度長く伸びるエンベロープにします。モジュレータになるオペレータ1では立ち上がりだけ倍音が多く、あとは一定にして音色変化はさせません。音量調整としてオペレータ2についてもキースケーリングをかけます。また、高音部にかけて音のシャープさを出すためには、RATEのスケーリングを行うとよいでしょう。

4. データの再調整

以上で音作りはほぼ終了ですが、EGなどのセッティングにより音色が幾分違ったものになってきます。この場合、オペレータの出力レベルやフィードバックレベルを再調整して、最終的な音に仕立てます。例えば、金属的な響きが強すぎると思われる場合には、オペレータ1のレベルを下げます。

5. エフェクト付け

最後にエレクトリックピアノの音をより生かすために、トレモロ効果をLFOによってつけ加えます。これは内蔵の振幅変調の機能を利用してもよいですし、ソフトウェアでTOTAL LEVELの値を2〜6Hzの周期で更新（三角波で可）することも可能です。

2. トランペット

1. オペレータ出力

モジュレータであるオペレータ1のトータルレベルは$10〜$28程度の控えめな値にし、フィードバックレベルはブライトな響きを出すために最大の「7」にします。

オペレータの周波数

基本的には、両方のオペレータ共に1倍にセットすればよいでしょう。

2. EG

2つのオペレータとも、ゆっくりとしたアタック音にします。そしてブラスのサウン

ドではモジュレータのアタックはすべてキャリアよりも遅くします。「ブァン」というブラス特有のアタックを表現するのに必要なことです。

3. キースケーリング

ゆっくりとした立ち上がりにエンベロープをセットしたため、高音部でハギレが悪くなります。このため、速いパッセージを弾いたときに不自然にならないように、レイトスケーリングを少しかけます。

4. LFO

ブラスはどんな上手なプレーヤーが吹いても、ロングトーンの場合にはピッチがほんの少し揺れてきます。これを表現するためにビブラート効果を加えます。

3. リズム音について

リズム音は7、8、9のチャンネルを使って作られます。この3チャンネル6スロットで計5音のリズム音を作るわけですが、バスドラム(BD)のみは2スロットでFM音を作ります。ここでは、残りの4音（ハイハット、トップシンバル、タム、スネアドラム）について説明します。

OPLLには、リズム楽器のためにホワイトノイズジェネレータと数種の周波数を合成して得られるノイズ発振器があります。このノイズ発振器は8チャンネルと9チャンネルの周波数情報(BLOCK、F-Number、MULTIPLE)より作られ、ホワイトノイズと合成することにより各リズム楽器に適した位相出力を発生して、オペレータに渡します。つまり、ここでは2つの周波数情報から4つの楽器の位相を作っていることになります。

なお、2つの設定周波数は経験的に3:1 (f8CH = 3×f9CH)が良いとされています。これで、各楽器の位相データが得られたことより、この出力にエンベロープの情報を掛け合わせます。エンベロープは1スロットに1リズム楽器と設定されているため、メロディ楽器同様各リズム楽器の特徴をつかんだ値が音色ROMにセットされています。

4章 MSX-AUDIO

4.1 ハードウェア

4.1.1 基本構成

1. 最小構成

■音源LSI	MSX-AUDIO LSI（Y8950）
■ DAC LSI	YM-3014
■ ADPCM/PCMデータ用RAM	256Kビット
■プログラム用ROM	128Kバイト
■プログラム用RAM	4Kバイト
■入出力端子	ミュージックキーボード接続端子 音声入力端子（マイクレベル ミニジャック） 音声出力端子（ラインレベル RCAジャック）

2. 最大構成

■音源LSI	MSX-AUDIO LSI（Y8950）×2
■ DAC LSI	DAC LSI（YM-3014）×2
■ ADPCM/PCMデータ用RAM	256K×8ビット
■ ADPCM/PCMデータ用ROM	256K×8ビット
■プログラム用ROM	128Kバイト
■プログラム用RAM	4Kバイト
■入出力端子	ミュージックキーボード接続端子 音声入力端子（マイクレベル ミニジャック） 音声出力端子（ラインレベル RCAジャック）

4.1.2 I/Oの構成

表7.43 MSX-AUDIO I/O構成

I/Oアドレス	Read/Write	MSX-AUDIOレジスタ名
C0H	W	アドレスレジスタ*1
C1H	R/W*3	データレジスタ*1
C2H	W	アドレスレジスタ*2
C3H	R/W*3	データレジスタ*2
C4H ⋮ C7H		システム予約

注意
*1 マスターチャンネル（必須）
*2 スレーブチャンネル（オプション）
*3 リード可能なレジスタのみ

また、MSX-AUDIOはY8950が持っているI/Oポートも使用します。

表7.44 MSX-AUDIOの4ビット汎用入出力ポート

ビット	意　味	
3	録音再生時のフィルタ切り換え（出力）	
	0	ADPCM/PCM音用フィルタ
	1	FM音用フィルタ
2	立ち上がり時のプログラム指定スイッチ（入力）	
	0	拡張BASIC
	1	内蔵アプリケーション
1	内蔵アプリケーション使用（出力）	
0	内蔵アプリケーション使用（出力）	

4.1.3 メモリの構成

MSX-AUDIOではスロットの管理とは別にローカルなバンクを使用しています。バンクは4つあり1つのバンクは32Kバイトです。MBIOSと拡張BASIC動作時、内蔵シンセサイザ動作時

およびADPCMデータROM読みだし時にバンクを切り替えます。バンクは3FFEH番地にメモリマップされたI/Oポートに書き込むことで切り替わります。そのビット構成は以下のとおりです。このポートはどのバンクが表に出ていても常にアクセスできます。また7FFEH番地からも書き込むことができます。

表7.45 ローカルバンクのI/O構成

ビット	意 味	
7〜2	使用していません。	
1と0	00	バンク0 MBIOS、拡張BASIC
	01	バンク1 内蔵アプリケーション
	10	バンク2 ADPCMデータROM1
	11	バンク3 ADPCMデータROM2

注 意　リセット時は必ず「00」にする。

また、この他にメモリマップされたI/Oポートがあります。これはY8950をポートに割り付けるためのポートです。アドレスは3FFFHです。このポートもどのバンクが表に出ていても常にアクセスできます。また、7FFFH番地からも書き込むことができます。拡張BASICはリセット時の初期化で、Y8950をポートに割り付けます。

表7.46 メモリマップドのI/O構成

ビット	意 味	
7〜2	使用していません。	
1と0	00	Y8950は割り付けられない。
	01	Y8950はC0HからのI/Oアドレスに割り付けられる。
	10	Y8950はC2HからのI/Oアドレスに割り付けられる。
	11	2つのY8950が1つのカートリッジにあり、それぞれC0HとC2Hに割り付けられる。

注 意　リセット時は必ず「00」にする。

上記2つのI/Oポートは書き込み時においてのみ選択され、読み出し時は通常のメモリが読まれなければなりません。

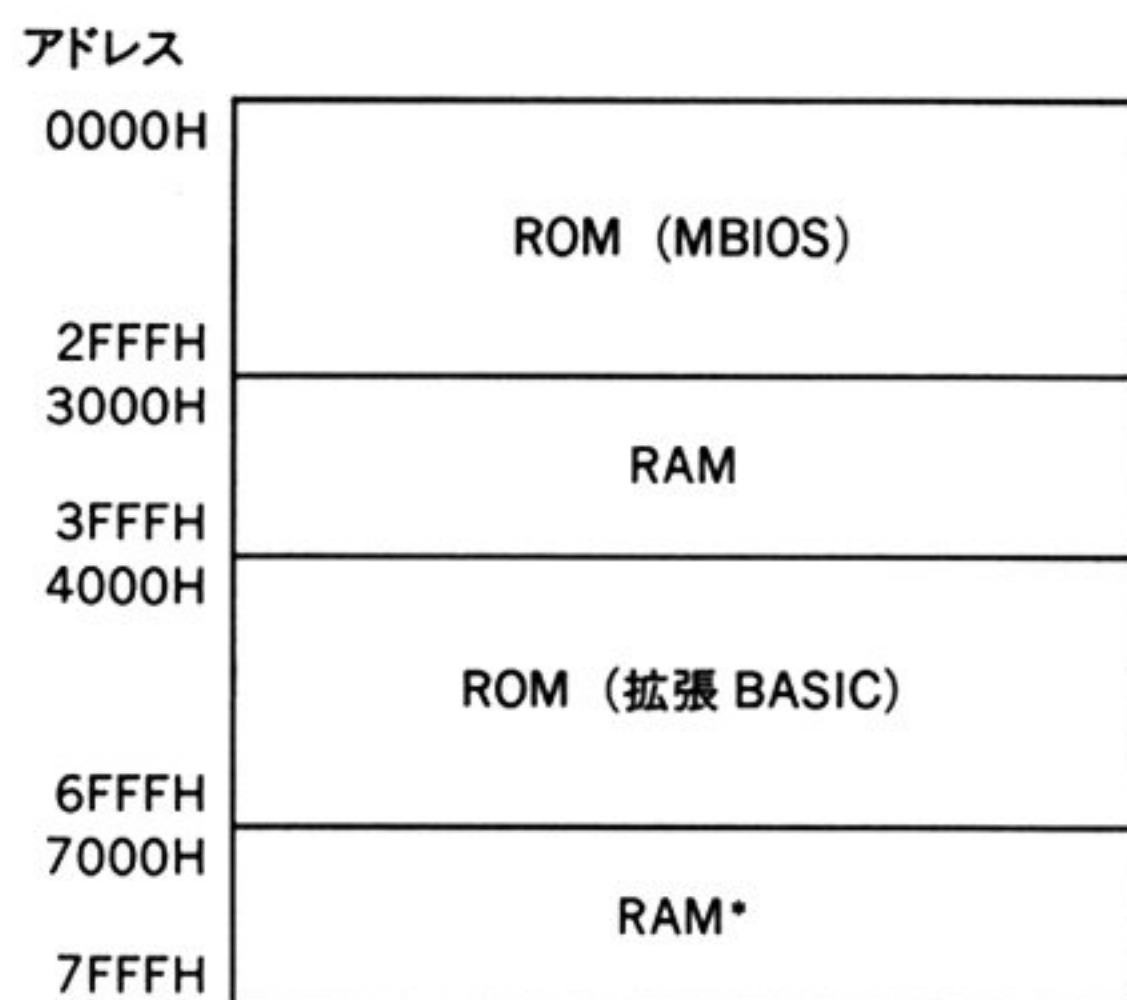
アドレス	
0000H〜2FFFH	ROM (MBIOS)
3000H〜3FFFH	RAM
4000H〜6FFFH	ROM (拡張 BASIC)
7000H〜7FFFH	RAM*

*7000H〜7FFFH の RAM は 3000H〜3FFFH の RAM のイメージです。

図7.28　バンク0 (MBIOS、拡張 BASIC 動作時)

アドレス	
4000H〜BFFFH	ROM (内蔵アプリケーション)

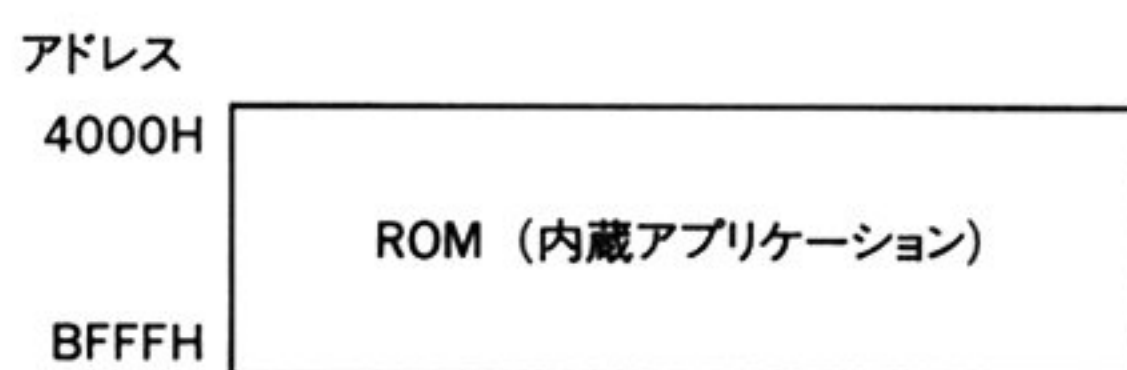



図7.29　バンク1 (内蔵アプリケーション動作時)

アドレス	
4000H〜BFFFH	ADPCM データ ROM1

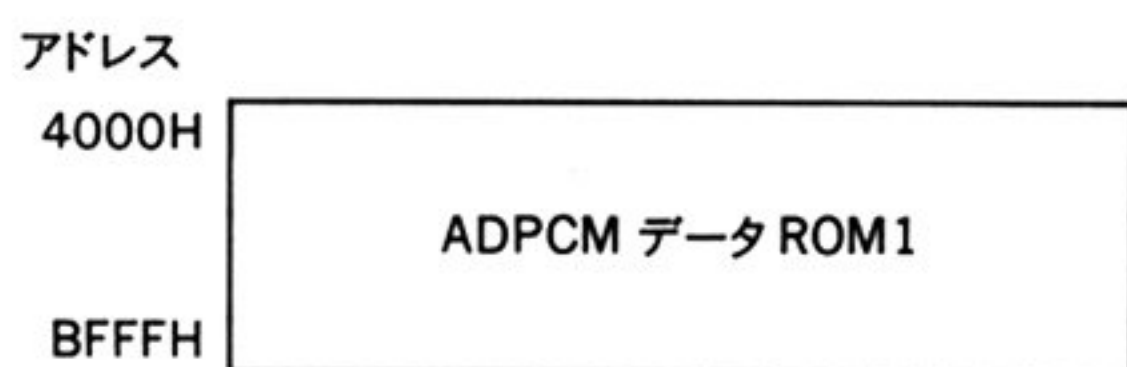



図7.30　バンク2 (ADPCM データ ROM1 読み出し時)

アドレス	
4000H〜BFFFH	ADPCM データ ROM2

図7.31　バンク3 (ADPCM データ ROM2 読み出し時)

4.1.4 ミュージックキーボードスキャンポート

1. キーボードマトリックス

表7.47　ミュージックキーボートのマトリックス

		入力ポート ビット番号							
		7	6	5	4	3	2	1	0
出力ポートビット番号	7	−	C	B	A#	−	A	G#	G
	6	−	F#	F	E	−	D#	D	C#
	5	−	C	B	A#	−	A	G#	G
	4	−	F#	F	E	−	D#	D	C#
	3	−	C	B	A#	−	A	G#	G
	2	−	F#	F	E	−	D#	D	C#
	1	−	C	B	A#	−	A	G#	G
	0	C	F#	F	E	−	D#	D	C#

ミュージックキーボードの最も左のキーが出力0番の入力7番でその1つ右側のキーが出力0番の入力0番です。最も右のキーが出力7番の入力6番です。

4.1.5 MSX-AUDIOを直接アクセスする場合の注意

この章で公開された内容を利用すれば、I/OポートからMSX-AUDIOを直接アクセスして音を鳴らすことができます。しかし、MSX-AUDIOのレジスタへデータを書き込む場合、タイミングによっては、正常に動作しなくなる可能性があります。したがって、商用のアプリケーションソフトウェアは直接MSX-AUDIOをアクセスしてはいけません。MSX-AUDIOにアクセスする場合は、必ず、MSX-AUDIO拡張BIOSかMSX-AUDIO MBIOSを使用するようにして下さい。

1. ウエイト

MSX-AUDIOでは、内部レジスタにアドレスやデータを書き込むと、次の動作に移るまでにはウエイト時間が必要です。このウエイト時間は、レジスタのアドレスの指定とデータの書き込みで異なります。表7.48で指定された時間だけ、CPUはMSX-AUDIO LSIに対して、次の動作を

待たなければなりません。

このウエイト時間を無視した場合は、そのときに設定したデータは保証されません。

表7.48 ウエイト時間

モード	ウエイト時間
アドレス指定	3.36μsec
データ書き込み	23.52μsec (3.36μsec)

注 意 データ書き込み時の（ ）の数値は$00～$0Aまでのレジスタに適用します。

4.2 拡張BASIC

4.2.1 概要

MSX-AUDIOには、各機能を簡単に使用できるように、MSX-AUDIO拡張BASICが用意されています。使い方はCALL AUDIOのように拡張ステートメントの形式です。CALLは_（アンダーバー）で代用できます。

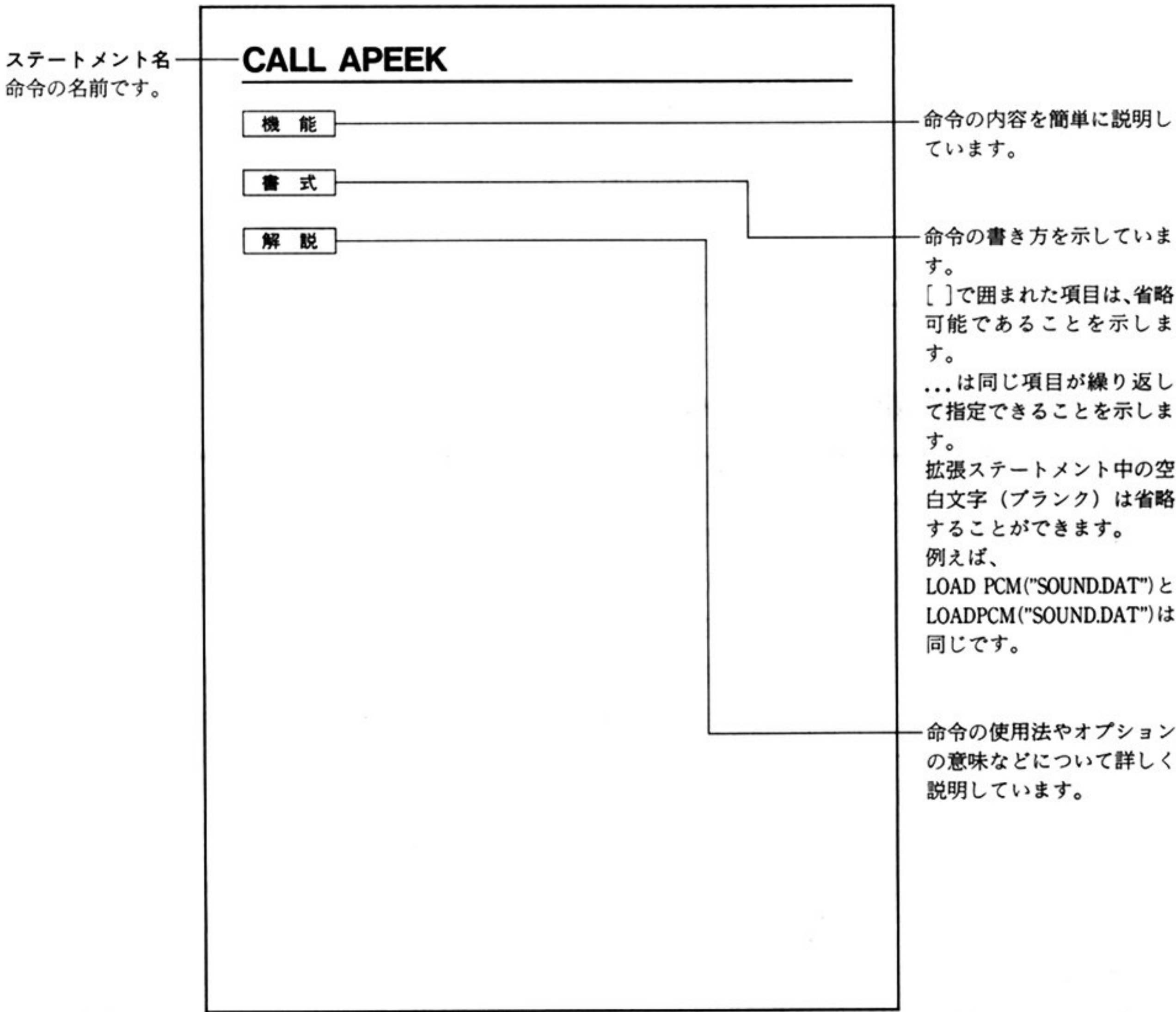
4.2.2 表記法

ステートメント名
命令の名前です。

CALL APEEK

機 能 ― 命令の内容を簡単に説明しています。

書 式 ― 命令の書き方を示しています。
[ ]で囲まれた項目は、省略可能であることを示します。
...は同じ項目が繰り返して指定できることを示します。
拡張ステートメント中の空白文字（ブランク）は省略することができます。
例えば、
LOAD PCM("SOUND.DAT")と
LOADPCM("SOUND.DAT")は
同じです。

解 説 ― 命令の使用法やオプションの意味などについて詳しく説明しています。

4.2.3 拡張 BASIC コマンド一覧

1. 拡張ステートメント(CALL 文と共に使用します)

コマンド名	機　能	ページ
APEEK	MSX-AUDIO のシステムメモリを参照します。	286
APOKE	MSX-AUDIO のシステムメモリを変更します。	286
AUDIO	MSX-AUDIO システムを初期化します。	287
AUDREG	MSX-AUDIO LSI のレジスタに値を書き込みます。	289
BGM	バックグラウンド処理を行うかどうかを指定します。	289
PITCH	FM 音源の楽音の音高(ピッチ)を与えます。	290
PLAY	音楽をミュージックマクロランゲージにしたがって演奏します。	291
PLAY	PLAY 文が音楽を演奏中かどうかを返します。	294
STOPM	バックグラウンドで実行中の PLAY 文の演奏、ADPCM、MK 記録再生を停止します。	295
SYNTHE	アプリケーションプログラムを呼び出します。	295
TEMPER	音律(テンペラメント)を与えます。	296
TRANSPOSE	FM 音源の楽音に対してセント単位で移調を与えます。	297
VOICE	FM音源の各チャンネルに音色(ボイス)を直接に設定します。	298
VOICE COPY	音色パラメータデータの転送を行ないます。	302

2. ADPCM / PCM 関係のステートメント

コマンド名	機　能	ページ
CONVA	PCM 形式のデータを ADPCM 形式のデータに変換します。	303
CONVP	ADPCM 形式のデータを PCM 形式のデータに変換します。	304
COPY PCM	ADPCM、PCM データを転送します。	304
LOAD PCM	ADPCM / PCM 音声ファイルをディスクからロードする。	306
PCM FREQ	ADPCM の外部 ROM / RAM を使用するローカルモード再生実行中にサンプリング周波数を与えます。	307
PCM VOL	ADPCM / PCM 再生の音量を設定します。	307
PLAY PCM	ADPCM / PCM 音声ファイルを再生します。	308
REC PCM	ADPCM / PCM により音声を音声ファイルに録音します。	308
SAVE PCM	ADPCM / PCM 音声ファイルをディスクへセーブする。	310
SET PCM	ADPCM / PCM の音声ファイルを初期設定します。	311

3. インスツルメント関係のステートメント

コマンド名	機　能	ページ
INMK	ミュージックキーボードの変化を報らせます。	314
KEY ON/KEY OFF	インスツルメントにキーオン、オフを与えます。	315
MK PCM	インスツルメントとして演奏するADPCMの音のファイル番号を指定します。	315
MK TEMPO	ミュージックキーボード演奏記録、再生とメトロノーム機能の速度を与えます。	316
MK VEL	インスツルメントにベロシティを設定します。	317
MK VOICE	インスツルメントのボイス（音色の種類）を設定します。	317
MK VOL	インスツルメントの音量を設定します。	318

4. MK（ミュージックキーボート）記録関係のステートメント

コマンド名	機　能	ページ
APPEND MK	MK記録の追加記録を行ないます。	320
CONT MK	STOPM文により停止したMKの記録、再生を再開します。	320
MK STAT	MK記録システムの状態を報らせます。	321
PLAY MK	MK記録の再生を行ないます。	321
REC MK	インスツルメントの演奏の記録を行ないます。	322
RECMOD	MK記録の記録モードを設定します。	323

4.2.4 拡張 BASIC の解説

1. 拡張ステートメント

CALL APEEK

機 能

MSX-AUDIO のシステムメモリの指定された番地の内容を読み出します。

書 式

CALL APEEK(<アドレス>, <数値変数>)

文 例

```
CALL APEEK(&H7000, A):PRINT A
```

解 説

<アドレス>で指定した MSX-AUDIO のシステムメモリの内容を読み出し、<数値変数>に代入します。
MSX-AUDIO のシステムメモリは 0000H～7FFFH に置かれています。この部分へのアクセスはスロットが異なっているため通常の PEEK 文では行なえません。CALL APEEK 文はそれを行なうためのステートメントです。8000H～FFFFH がアドレスとして指定された場合には、BASIC の使用しているメモリ空間をアクセスするので、通常の PEEK 文と同じ働きをします。

CALL APOKE

機 能

MSX-AUDIO のシステムメモリの指定された番地にデータを書き込みます。

書 式

CALL APOKE(<アドレス>, <データ>)

文 例

```
CALL APOKE(&H7000, 0)
```

解　説

＜アドレス＞で指定したMSX-AUDIOのシステムメモリの番地に＜データ＞を書き込みます。
MSX-AUDIOのシステムメモリは0000H～7FFFHに置かれています。この部分へのアクセスはスロットが異なっているため通常のPOKE文では行なえません。CALL APOKE文はこれを行なうためのステートメントです。8000H～FFFFHがアドレスとして指定された場合には、BASICの使用しているメモリ空間をアクセスしますので、通常のPOKE文と同じ働きをします。

CALL AUDIO

機　能

MSX-AUDIOシステムを初期化します。

書　式

CALL AUDIO[(＜モード＞[, ＜インスツルメントへのチャンネル数＞[, ＜PLAY文第1文字列へのチャンネル数＞[, ＜PLAY文第2文字列へのチャンネル数＞[, ・・・[, ＜PLAY文第9文字列へのチャンネル数＞]]]]]]]]]])]

文　例

CALL AUDIO
デフォルトの設定をする。

CALL AUDIO(0, 9)
すべてのチャンネルをインスツルメントに割り当てる。

CALL AUDIO(0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1)
1チャンネルずつPLAY文の文字列に割り当てる。

解　説

MSX-AUDIO LSIの初期化とともに、9個のFM音源のチャンネルをどのように使用するかを指定します。AUDIO文により初期化を行うまでは、すべての拡張BASICステートメントは使用できません。
＜モード＞は0～7で、次のようなビットマップでMSX-AUDIOの動作モードを指定します。

bit0　リズム音を使用する。
bit1　PLAY文がPCM音源用文字列を使用する。
bit2　MSX-AUDIOをCSMモードに設定する。

MSX-AUDIOをCSMモードに設定すると、すべてのFM音源(リズム音を含む)に対するコントロールは無効になります。
リズム音を使用する場合には、チャンネル7、8、9を使うので、楽音に使えるのは残り6チャンネルになります。したがって、インスツルメントへのチャンネル数とPLAY文で使用するチャンネル数の和との総和は、リズム使用時には6以下、リズムを使わないときは9以下になります。
チャンネルの使用割り当ては、PLAY文では、チャンネル番号の小さい方(1、2、3・・・)から、インスツルメントではチャンネル番号の大きい方(9、8、7・・・、リズム使用時には6、5、4・・・)から割り当てます。
パラメータを1つ以上指定した場合、他のパラメータの省略時の値は0となります。
PLAY文の文字列へのチャンネル数を0に設定したり、途中のパラメータを省略することはできません。次の例を参照して下さい。

```
CALL AUDIO(0,3,0,5,0)
               ↑ ↑
```

0を設定してはいけない(「Illegal function call」になります)

```
CALL AUDIO(0,2,1, ,2)
                ↑
```

省略してはいけない(「Syntax error」になります)

パラメータなしで使われたときは、

```
CALL AUDIO(1, 0, 2, 2, 2)
```

と同じになります。すなわち、

- FM音源のチャンネル1〜2をPLAY文の最初の文字列に割り当てる。
- FM音源のチャンネル3〜4をPLAY文の2番目の文字列に割り当てる。
- FM音源のチャンネル5〜6をPLAY文の3番目の文字列に割り当てる。
- FM音源のチャンネル7〜9をリズム音に使用する。
- PLAY文の5番目以降7番目の文字列はPSG音源の制御に割り当てる。
- インスツルメントにはチャンネルを割り当てない。
- PCM音源はPLAY文では扱わない。
- CSMモードは使用しない。

という意味になります。
AUDIO文を実行すると、システムの割り込みのフックがMSX-AUDIOのシステムソフトウェアにリンクされるので、割り込み処理ルーチンのオーバーヘッドが増え、システムのスループットが低下します。特に、ミュージックキーボードを使用する文を実行するとその影響が大きくなります。

CALL AUDREG

機　能

MSX-AUDIO LSIのレジスタに値を書き込みます。

書　式

CALL AUDREG(＜レジスタ番号＞, ＜データ＞[, ＜チャンネル番号＞])

文　例

CALL AUDREG(&HBD, 0)

解　説

＜レジスタ番号＞で指定したMSX-AUDIO LSIのレジスタに対して、＜データ＞を書き込みます。
＜チャンネル番号＞は0または1で、省略時は0です。0の場合には、第1チャンネルのMSX-AUDIO LSIに、1の場合には第2チャンネルのMSX-AUDIO LSIにアクセスします。
システムソフトウエアが割り込みなどでひんぱんに書き込んでいるレジスタには効果がない場合やシステムの再起動が必要になる場合があります。

CALL BGM

機　能

バックグラウンド処理を行うかどうかを指定します。

書　式

CALL BGM(＜変数＞)

文　例

CALL BGM(0)

バックグラウンド処理を行なわない。

CALL BGM(1)

バックグラウンド処理を行なう。

解　説

<変数>は0または1の値で、次のような意味を持ちます。

0　　バックグラウンド処理を行なわない。

1　　バックグラウンド処理を行なう。

AUDIO文による初期化ではバックグラウンド処理が指定されていますが、<変数>に0を指定することでフォアグラウンド処理にすることができます。
次にあげる機能はバックグラウンドで処理することができます。

- PLAY文による演奏。
- 外部メモリを使用するローカルモードのADPCMの録音再生。
- 配列を記録領域に使用しないMK記録の記録再生。

CALL PITCH

機　能

FM音源の楽音の音高（ピッチ）を与えます。

書　式

CALL PITCH(<ピッチ1>[, <ピッチ2>])

文　例

CALL PITCH(440)

解　説

FM音源で発生する楽音の音高を指定します。<ピッチ>の範囲は410～459で単位は[Hz]です。中央Cのすぐ上のA音(a2)の周波数で音高を表わします。トランスポーズとは独立に設定でき、初期値は440です。
ピッチ(またはトランスポーズ値)を変えると、リズム音や音程をもたない音を除くFM音の音の高さが変化します。PCM音源やPSG音源には作用しないので注意して下さい。

パラメータを2つ取る場合の<ピッチ1>は第1チャンネルの、<ピッチ2>は第2チャンネルのピッチを設定します。
パラメータを1つしか取らない場合で、2つのMSX-AUDIO LSIが実装されている場合には、<ピッチ1>が両方のチャンネルに有効となります。
トランスポーズについてはTRANSPOSE文の項を参照して下さい。

PLAY

機 能

音楽をミュージックマクロランゲージにしたがって演奏します。

書 式

PLAY[#<モード>,]<文字列1>[, <文字列2>[, <文字列3>]・・・[, <文字列13>]

文 例

PLAY # 2, "CD", "EF", "GA"

解 説

PLAY文は音楽を演奏するもので、FM音源(9)、PCM音源(1)、PSG音源(3)の最大13声まで同時発声できます。<文字列>に書かれたミュージックマクロランゲージ(MML)にしたがって演奏します。
他の拡張命令と異なりCALL文は必要ありません。
<モード>の設定は次のとおりです。

モード	意 味
0(省略)	PSGのみが音源となり、文字列は最大3つまでです。従来のPLAY文と互換性があります。
1	演奏データをMIDI*ポートに出力します。
2、3	FM音源、リズム音、PCM音源、PSG音源を使用できます(2のときと3のときで動作に違いはありません)。

<文字列>と音源との関係は始めから順に、
<FM音源用文字列1>、・・・, <FM音源用文字列n>、
<PCM音源用文字列>、<リズム音用文字列>、
<PSG音源用文字列1>、<PSG音源用文字列2>、
<PSG音源用文字列3>

となります。nはAUDIO文で設定されたミュージックマクロランゲージの個数です。

CALL AUDIO文でリズム音やPCM音源を使用しないモードに設定した場合は、リズム音用文字列やPCM音源用文字列をカンマ(,)と共に省略しなければいけません。

2つのMSX-AUDIO LSIが実装されている場合には、両方のチャンネルに同じ効果をもちます。

例としてデフォルトのAUDIO文に対する文字列の配列をあげると次のようになります。

<FM音源用文字列1>、<FM音源用文字列2>、<FM音源用文字列3>、<リズム音用文字列>、<PSG音源用文字列1>、<PSG音源用文字列2>、<PSG音源用文字列3>

| 注　意 | *MIDIの機能はオプションで、別途サポートされているときのみ有効です。

ミュージックマクロランゲージ(MML)の仕様

表7.49　FM音源、PSG音源、PCM音源用MMLの仕様一覧

文　字	意　味	値のとる範囲	初期値
Mn	エンベロープ周期を設定する	1≦n≦65535	M255
Sn	エンベロープ形状を設定する	0≦n≦15	S0
Vn	音量を設定する	0≦n≦15	V8
Ln	長さを設定する	1≦n≦64	L4
Qn	音の長さの割合を設定する	1≦n≦8	Q8
On	オクターブを設定する	1≦n≦8	O4
>	オクターブを1つ上げる		
<	オクターブを1つ下げる		
Tn	テンポを設定する	32≦n≦255	T120
Nn	nで指定された高さの音を発生する	0≦n≦96	
Rn	休符を設定する	1≦n≦64	R4
A～G	音程を発生する		
+、#	音を半音上げる		
−	音を半音下げる		
.(ピリオド)	音符や休符の長さを1.5倍する		
=x;	パラメータnを変数xで設定する		
Xx;	文字変数xに入っているMMLを演奏する		
&	タイ、前後の音をつなぐ		
{ }n	連符、n分音符を{ }の中の音程の個数で等分にした音を発生する	1≦n≦64	Lnで設定された値
@n	n番の音色に切り換える	0≦n≦63	
@Vn	音量を細く設定する	0≦n≦127	
@Wn	nで指定された長さだけ状態を継続する	1≦n≦64	Lnで設定された値

文　字	意　味	値のとる範囲	初期値
Yr, d	MSX-AUDIO LSI のレジスタ r に d を書き込む		
Zd*	MIDI にデータ d を送る		

注　意　*MIDI 機能はオプションで、別途サポートされているときのみ有効です。

リズム音用 MML の仕様

リズム音の場合、1 つの MML で同時にいくつかの音を発生するため楽音用とは異なった記述様式をとります。まず、鳴らしたい楽器を並べてその後に長さを指定します。

表 7.50　リズム音用 MML の仕様一覧

文字	意　味	値のとる範囲	初期値
B	バスドラム音を発生する		
S	スネアドラム音を発生する		
M	タムタム音を発生する		
C	シンバル音を発生する		
H	ハイハット音を発生する		
!	直前の楽器の音量をアクセントボリュームにする		
n	直前までに書かれた楽音を発生し､n 分音符分待つ	1≦n≦64	
Vn	アクセントの付いていない楽音の音量を設定する	0≦n≦15	8
@An	アクセントの付いている楽音の音量を設定する	0≦n≦15	

Tn、@Vn、Rn、= x ；、Xx ；､. は FM 音源用と同じです。

例

"BSH8H8S!H8H8"

- ■バス、スネア、ハイハットを鳴らし、8 分音符分待ちます。
- ■ハイハットを鳴らし、8 分音符分待ちます。
- ■スネアをアクセント付でハイハットと鳴らし 8 分音符分待ちます。
- ■ハイハットを鳴らし、8 分音符分待ちます。

表 7.51 MML と各音源との対応一覧

文字	FM 音源	PSG 音源	PCM 音源
Mn	*1	○	*1
Sn	*1	○	*1
Vn	○	○	○
Ln	○	○	○
Qn	○	*1	○
On	○	○	○
>	○	○	○
<	○	○	○
Tn	○	○	○
Nn	○	○	○
Rn	○	○	○
A~G	○	○	○
+, #	○	○	○
−	○	○	○
.	○	○	○
= x ;	○	○	○
Xx ;	○	○	○
&	○	○	○
{} n	○	*3	○
@n	○	*1	○
@Vn	○	*1	○
@Wn	○	*2	○
Yr, d	○	*1	○
Zd	○	*1	○

注 意

*1 無視されます。

*2 Rn と同じです。

*3 PSG 音源に対しては使用できません。使用するとエラーになります。

CALL PLAY

機 能

PLAY 文が音楽を演奏中かどうかを返します。

書 式

CALL PLAY(<PLAY 文のストリング番号>, <変数名>)

文 例

CALL PLAY(0, A):PRINT A

解 説

PLAY文のミュージックキューの状態を調べ、各チャンネルが音楽を演奏中かどうかを判断し、演奏中であれば「−1」、そうでなければ「0」の値を返します。ただし、<ストリング番号>として0が与えられた場合は、いずれかのストリングが演奏中であれば「−1」を、そうでなければ「0」を返します。
PLAY文のストリング番号は、CALL AUDIO文で指定したストリング数+3まで使えます。すなわち、AUDIO文で指定したFM音源、PCM音源に加え3チャンネルのPSG音源について有効です。

CALL STOPM

機 能

バックグラウンドで実行中のPLAY文の演奏、ADPCM、MK記録再生を停止します。

書 式

CALL STOPM[(<変数名>)]

文 例

CALL STOPM

解 説

バックグラウンドで実行中のPLAY文の音楽の演奏、外部メモリを使用したADPCMの録音、再生、MK記録再生を停止します。
パラメータとして<変数名>を与えると、MK記録・再生の中止されたアドレスの次のアドレス(CONT MK文により再開されるアドレス)が返されます。

CALL SYNTHE

機 能

MSX-AUDIO内蔵のアプリケーションプログラムを呼び出します。

書 式

CALL SYNTHE

文 例

CALL SYNTHE

解 説

この命令の実行は、CALL AUDIO 文が実行される以前でなければなりません。

CALL TEMPER

機 能

音律（テンペラメント）を与えます。

書 式

CALL TEMPER(<音律番号>)

文 例

CALL TEMPER(0)

解 説

音律を与えるステートメントで、FM 音源の楽音の音高に影響を与えます。指定できる<音律番号>は 0～21 です。音律は 1 オクターブをどのような比率で 12 音に分割するかを決めるもので、古典音楽には古典音律が適していると言われます。初期値は 9 番の完全平均律です。

番 号	音 律	
0	ピタゴラス	
1	ミーントーン	
2	ヴェルクマイスター	
3	ヴェルクマイスター（修正）	
4	ヴェルクマイスター（別）	
5	キルンベルガー	
6	キルンベルガー（修正）	
7	ヴァロッティ・ヤング	
8	ラモー	
9	完全平均律（デフォルト）	
10	純正律 c	メジャー（a マイナー）
11	純正律 cis	メジャー（b マイナー）

番　号	音　律	
12	純正律 d	メジャー（h マイナー）
13	純正律 es	メジャー（c マイナー）
14	純正律 e	メジャー（cis マイナー）
15	純正律 f	メジャー（d マイナー）
16	純正律 fis	メジャー（es マイナー）
17	純正律 g	メジャー（e マイナー）
18	純正律 gis	メジャー（f マイナー）
19	純正律 a	メジャー（fis マイナー）
20	純正律 b	メジャー（g マイナー）
21	純正律 h	メジャー（gis マイナー）

（修正）は平島達司氏による。

CALL TRANSPOSE

機　能

FM 音源の楽音に対してセント単位で移調を与えます。

書　式

CALL TRANSPOSE(<トランスポーズ値1>[, <トランスポーズ値2>])

文　例

```
CALL TRANSPOSE(0)
CALL TRANSPOSE(0, 700)
```

解　説

移調を行なうためのステートメントで、単位はセントです。これは、半音を100とした移調の単位で、1オクターブ上げるには、+1200を指定します。

トランスポーズ値として許される範囲は、±12799以内ですが、実際にはFM音源の音色によって、ある高さの範囲以外は制限されます。音高精度はLSIの制限により±2セント程度です。

トランスポーズはピッチとは独立して設定できます。AUDIO文による初期化の値は0です。ピッチについてはCALL PITCH文を参照して下さい。

パラメータを2つ取る場合、<トランスポーズ値1>は第1チャンネルの、<トランスポーズ値2>は第2チャンネルのピッチを設定します。

パラメータを1つしか取らない場合で、2つのMSX-AUDIO LSIが実装されている場合には、<トランスポーズ値1>が両方のチャンネルに対して有効になります。

CALL VOICE

機 能

FM音源の各チャンネルに音色（ボイス）を直接設定します。

書 式

CALL VOICE([<チャンネル1用のボイス>],[<チャンネル2用のボイス>], ‥‥,
[<チャンネル9用のボイス>])

ボイス=@+単純変数　または
=配列変数名

文 例

```
CALL VOICE(@0, @0, @0, , , , @7, @7, @7)
CALL VOICE(@0, @0)
```

解 説

MSX-AUDIOの9チャンネルあるFM音源のそれぞれに音色を設定します。
音色の設定方法には2つあります。システムに備えられている音色ライブラリを使う場合には、0～63の音色の番号を単純変数または定数により指定します。この場合には、変数名または定数の前に@記号をつけて、次で説明する配列変数名と区別します。
プログラムにより音色パラメータを与えて設定する場合には、配列変数に音色パラメータを入れてその配列変数名を指定します。音色パラメータのフォーマットの詳細はVOICE COPY文の解説を参照してください。パラメータを省略したチャンネルの音色は変更されません。
インスツルメントに指定したチャンネルは、MK VOICE文でまとめて設定できます。MK VOICE文を参照して下さい。
MSX-AUDIO LSIが2つ実装されている場合には、同じパラメータが両方のチャンネルに設定されます。

システム音色ライブラリ一覧

音色番号0～31はROM内に置かれているため変更できませんが、音色番号32～63のものはVOICE COPY文により変更することができます。略号は音色の名前としてライブラリに登録されているものです。

表 7.52　システム音色ライブラリー覧

音色番号	音色名	略　号
0	Piano 1	Piano 1
1	Piano 2	Piano 2
2	Violin	Violin
3	Flute 1	Flute
4	Clarinet	Clarinet
5	Oboe	Oboe
6	Trumpet	Trumpet
7	Pipe Organ 1	PipeOrgn
8	Xylophone	Xylophon
9	Organ	Organ
10	Guitar	guitar
11	Santool 1	Santool
12	Electric Pianol	Elecpian
13	Clavicode 1	Clavicod
14	Harpsicode 1	Harpsicd
15	Harpsicode 2	Harpscd2
16	Vibraphone	Vibraphn
17	Koto 1	Koto
18	Taiko	Taiko
19	Engine 1	Engine
20	UFO	UFO
21	Synthesizer bell	SynBell
22	Chime	Chime
23	Synthesizer bass	SynBass
24	Synthesizer	Synthsiz
25	Synthesizer Percussion	SynPercu
26	Synthesizer Rhythm	SynRhyth
27	Harm Drum	HarmDrum
28	Cowbell	Cowbell
29	Close Hi-hat	ClseHiht
30	Snare Drum	SnareDrm
31	Bass Drum	BassDrum
32	Piano 3	Piano 3
33	Electric Piano 2	Elecpia 2
34	Santool 2	Santool 2
35	Brass	Brass
36	Flute 2	Flute 2
37	Clavicode 2	Clavicd 2
38	Clavicode 3	Clavicd 3
39	Koto 2	Koto 2

音色番号	音色名	略 号
40	Pipe Organ 2	PipeOrg 2
41	PohdsPLA	PohdsPLA
42	RohdsPRA	RohdsPRA
43	Orch L	Orch L
44	Orch R	Orch R
45	Synthesizer Violin	SynViol
46	Synthesizer Organ	SynOrgan
47	Synthesizer Brass	SynBrass
48	Tube	Tube
49	Shamisen	Shamisen
40	Magical	Magical
51	Huwawa	Huwawa
52	Wander Flat	WnderFlt
53	Hardrock	Hardrock
54	Machine	Machine
55	Machine V	MachineV
56	Comic	Comic
57	SE-Comic	SE-Comic
58	SE-Laser	SE-Laser
59	SE-Noise	SE-Noise
60	SE-Star 1	SE-Star
61	SE-Star 2	SE-Star 2
62	Engine 2	Engine 2
63	Silence	Silence

音色パラメータデータのフォーマット

音色パラメータは次の表のように1音色あたり32バイトのデータを使います。オペレータ0とオペレータ1のデータは同じものの繰り返しになっています。

表7.53　音色パラメータデータのフォーマット

オフセット	内　容
	ヘッダ
0〜7	音色名
8〜9	ボイス移調
10	bit0　アルゴリズム
	bit1〜3　フィードバック
	bit4　固定ピッチ
	bit5　AMD/PMDロード可能
	bit6　PMD
	bit7　AMD
11〜15	予約
	オペレータ 0
16	bit0〜3　MULT
	but4　KSL
	bit5　EG
	bit6　PM
	bit7　AM
17	bit0〜5　トータルレベル
	bit6〜7　レベルキースケール
18	bit0〜3　ディケイレート
	bit4〜7　アタックレート
19	bit0〜3　リリースレート
	bit4〜7　サスティンレベル
20	bit4〜7　ベロシティセンシビリティ（0〜8）
21〜23	予約
	オペレータ 1
24	bit0〜3　MULT
	bit4　KSL
	bit5　EG
	bit6　PM
	bit7　AM
25	bit0〜5　トータルレベル
	bit6〜7　レベルキースケール
26	bit0〜3　ディケイレート
	bit4〜7　アタックレート
27	bit0〜3　リリースレート
	bit4〜7　サスティンレベル
28	bit4〜7　ベロシティセンシビリティ（0〜8）
29〜31	予約

CALL VOICE COPY

機 能

音色パラメータデータの転送を行ないます。

書 式

CALL VOICE COPY(＜パラメータ 1＞, ＜パラメータ 2＞)

文 例

```
DIM A(128)
CALL VOICE COPY(*, A)
```

音色番号 32〜63 のパラメータを配列 A に転送する。

```
CALL VOICE COPY(@0, PIANO)
```

音色番号 0 のパラメータを配列「PIANO」に転送する。

```
CALL VOICE COPY(@0, @32)
```

音色番号 0 のパラメータを音色番号 32 に転送する。

解 説

配列と音色ライブラリ (0〜63) の間でのデータを転送します。転送は、＜パラメータ 1＞から＜パラメータ 2＞へ行います。

@と単純変数が指定されたときは、その変数の値で指定される音色番号の音色データが対象となります。

ソース (パラメータ 1) に指定される音色番号は 0〜63 までのすべての値を指定することができますが、ディスティネーション (パラメータ 2) に指定される音色パラメータは音色番号 32〜63 までの、RAM に置かれているユーザー音色ライブラリに限ります。

@記号がない場合の変数名は配列変数とみなされ、その内容が転送の対象になります。

1 つの音色パラメータは 32 バイトです。音色パラメータの詳細については音色パラメータデータのフォーマットを参照して下さい。

*は音色番号 32〜63 のユーザー音色ライブラリ全データを意味し、それを使うときは他方のパラメータは、1K バイト以上の長さをもつ配列変数名でなければなりません(32×32＝1024)。

音色パラメータを他のファイルシステム、例えば、ディスクファイルなどに転送するには、まず配列変数に転送しておいて、COPY 文によりその配列変数をディスクに転送します。

2. ADPCM/PCM関係のステートメント

ADPCM/PCM音のデータは音声ファイルという、番号(0〜15)で管理するファイルとして扱われます。このファイルが記録される場所をデバイスとよび、MSX-AUDIO LSIの外部メモリ以外にもメインRAM、VRAMへも直接録音、再生ができます。

外部メモリを使用する場合のADPCM録音再生は、バックグラウンドで行うができます。また、ADPCMとPCMとの間のデータの変換をする機能もあります。

CALL CONVA

機能

PCM形式のデータをADPCM形式のデータに変換します。

書式

CALL CONVA(<ソースファイル番号>, <デスティネーションファイル番号>)

文例

CALL CONVA(1, 2)

<ソースファイル番号>で指定されるPCMデータ全体をADPCMデータに変換して、<デスティネーションファイル番号>で指定される音声ファイルに格納します。

解説

PCM形式のデータをADPCM形式のデータに変換します。

<ソースファイル番号>と<デスティネーションファイル番号>は、同じ番号は指定できません。デスティネーションファイルの長さは1/2になりますので、ソースファイルの長さが奇数長のときは1/2して、整数部分のみの変換を行ないます。

CALL CONVA文では、ソースファイルのデータの形式がPCMであることをチェックし、異なっていると、「Illegal function call」になります。

デスティネーションファイルは外部RAM、メインRAMまたはVRAMのいずれかでなければなりません(内容の変更可能なデバイス)。また、ファイルのタイプと長さはメインRAM(デバイス=4)を使用している場合以外は変更されます。

サンプリング周波数は、ソース側に合わせられます。

CALL CONVP

機　能

ADPCM形式のデータをPCM形式のデータに変換します。

書　式

CALL CONVP(＜ソースファイル番号＞, ＜デスティネーションファイル番号＞)

文　例

CALL CONVP(1, 2)

＜ソースファイル番号＞で指定されるADPCMデータ全体をPCMデータに変換して＜デスティネーションファイル番号＞で指定される音声ファイルに格納します。

解　説

ADPCM形式のデータをPCM形式に変換します。
＜ソースファイル番号＞と＜デスティネーションファイル番号＞は、同じ番号は指定できません。デスティネーションファイルの長さはソースファイルの2倍になります。
CALL CONVP文では、ソースファイルのデータの形式がADPCMであることをチェックし、異なっていると、「Illegal function call」になります。
デスティネーションファイルは外部RAM、メインRAMまたはVRAMのいずれかでなければなりません(内容の変更可能なデバイス)。またファイルのタイプと長さはメインRAM（デバイス=4）を使用している場合以外は変更されます。また、サンプリング周波数は、ソース側に合わせられます。

CALL COPY PCM

機　能

ADPCM、PCMデータを転送します。

書　式

CALL COPY PCM(＜ソースファイル番号1＞, ＜デスティネーションファイル番号2＞[, ＜オフセット1＞][, ＜長さ＞][, ＜オフセット2＞])

文　例

CALL COPY PCM(1, 2)

ファイル 1 の全体をファイル 2 に転送します。

解　説

ADPCM、PCM 音声ファイル間でのデータの全コピーや部分的コピーします。パラメータの意味は図 7.32 のとおりです。
CALL COPY PCM 文では、ファイルの ADPCM / PCM タイプチェックを行いません。また、転送後のデスティネーションファイルの長さやタイプは変わりません。
パラメータは 2 つの音声ファイル番号以外は省略できます。省略時の値は<オセフット>は 0、<長さ>はソースファイルの最後までになります。
<ソースファイル番号 1>として「#」のついた値を使用すると、システムに組み込みの ADPCM データからのコピーが実行されます。

ソースファイル
ディスティネーションファイル
先頭
オフセット 1
長さ
先頭
オフセット 2

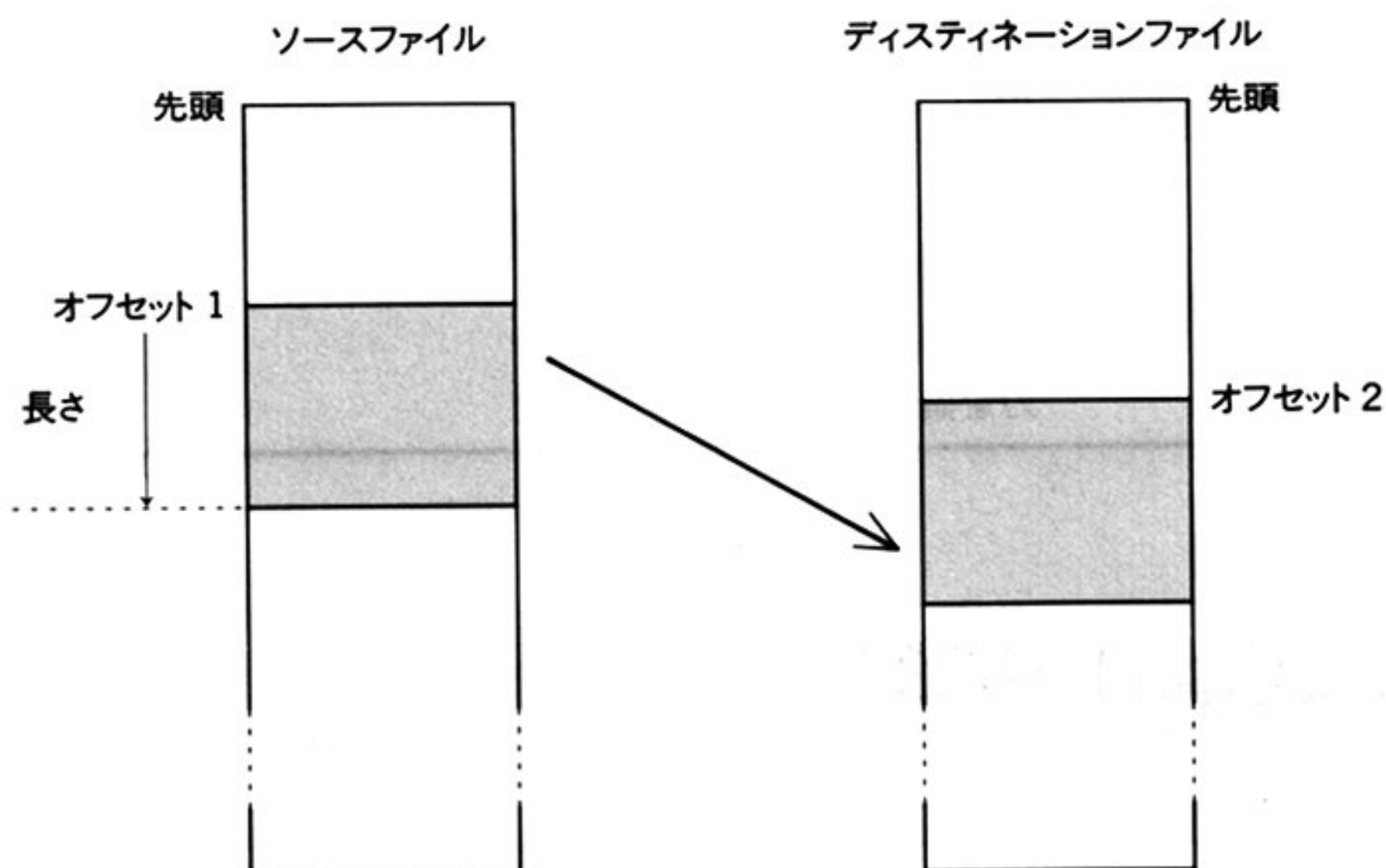



図 7.32　CALL COPY PCM 文のコピー範囲

■システム組み込み ADPCM 音声データ一覧

システム組み込みの ADPCM の音声データは表 7.54 のとおりです。これらのデータは CALL COPY PCM 文により 1 度音声ファイルに転送してから再生します。

表 7.54 システム組み込み ADPCM 音声データ一覧

ファイル番号	内　容	長さ（256バイト単位）
0	カッコー	35
1	ニワトリ	33
2	猫	20
3	犬	5
4	馬	29
5	ライオン	43
6	人の笑い声	26
7	ドアの閉まる音	11
8	ウィスキーを注ぐ音	4
9	靴音	1
10	行進	6
11	玩具	4
12	拍手	2
13	テニス	5
14	ゴルフのスイング	9
15	ゴルフのカップイン	9
16	刀を振る音1	6
17	刀を振る音2	7

CALL LOAD PCM

機　能

フロッピーディスクから ADPCM/PCM 音声ファイルをロードします。

書　式

CALL LOAD PCM(＜ファイル名＞，＜音声ファイル番号＞)

文　例

CALL LOAD PCM("DEMO.DAT", 1)

解　説

＜音声ファイル番号＞で指定された音声ファイルに＜ファイル名＞で指定されたフロッピーディスク上のファイルから ADPCM/PCM データを読み込みます。
SET PCM で設定した長さよりもファイルが長い場合は、指定された長さまでをロードします。音声ファイルのデータ形式（ADPCM/PCM）とサンプリング周波数は、ロードしたデータに合せます。

<ファイル名>は DISK　BASIC のファイルスペックに適合する文字列で、[ドライブ":"]＋最大 8 文字のファイル名＋[".''最大 3 文字の拡張子]です。
ファイル名としてデバイス名 (CON、PRN、LST、AUX、NUL) を指定すると、そのデバイスから入力されますが、入力途中に CTRL＋STOP または CTRL＋C などで中止したときは、次の PCM 関係のステートメントは正しく実行されません。その場合、次のステートメントの実行の前に CALL STOPM 文を実行してください。
CALL LOAD PCM 文は高速化のために、BASIC のフリーエリアを転送時のバッファとして使用します。したがって、フリーエリアが少ないときはロード時間が長くなります。

CALL PCM FREQ

機　能

ADPCM の外部 ROM・RAM を使用するローカルモード再生実行中にサンプリング周波数を与えます。

書　式

CALL PCM FREQ(<サンプリング周波数 1>[, <サンプリング周波数 2>])

文　例

CALL PCM FREQ(24000)

解　説

ローカルモードで ADPCM を再生中にサンプリング周波数を変更します。<サンプリング周波数>の値の範囲は 1800～49716 で、単位は Hz です。パラメータが 2 つのときは、<サンプリング周波数 1>が第 1 チャンネルに、<サンプリング周波数 2>が第 2 チャンネルになります。
MSX-AUDIO LSI が 2 つ実装されている場合で、パラメータが 1 つしかないときは、<サンプリング周波数 1>が両方のチャンネルに設定されます。

CALL PCM VOL

機　能

ADPCM / PCM 再生の音量を設定します。

書　式

CALL PCM VOL(＜ボリューム値1＞[, ＜ボリューム値2＞])

文　例

CALL PCM VOL(40)

解　説

ADCPM/PCM再生の音量を設定します。＜ボリューム値＞の範囲は0〜63で+6dB/8の変化をもちます。初期値はADPCMでは63、PCMでは32です。

パラメータが2つのときは、＜ボリューム値1＞が第1チャンネルに、＜ボリューム値2＞が第2チャンネルになります。

MSX-AUDIO LSIが2つ実装されている場合で、パラメータが1つしかないときは、＜ボリューム値1＞が両方のチャンネルに設定されます。

CALL PLAY PCM

機　能

ADPCM/PCM音声ファイルを再生します。

書　式

CALL PLAY PCM(＜音声ファイル番号＞[, ＜rep＞][, ＜オフセット＞][, ＜長さ＞][, ＜サンプリング周波数＞][, ＜チャンネル番号＞])

文　例

CALL PLAY PCM(0)

解　説

＜音声ファイル番号＞で指定されたADPCM/PCM音声ファイルの＜オフセット＞の位置から＜長さ＞のデータを＜サンプリング周波数＞で再生します。

パラメータ	値	意　味
＜サンプリング周波数＞	1800〜16000Hz	外部メモリ(デバイス番号0〜3)を使ってADPCM再生するとき(ローカルモード)に限り、1800〜49716Hzまで指定できます。
＜音声ファイル番号＞	0〜15	再生する音声ファイルを指定します。

<rep>	0、1	1を指定するとリピートモードになり、指定された区間の再生を繰り返します。
<チャンネル番号>	0、1	0は第1チャンネルへの再生、1は第2チャンネルへの再生です。ただし、外部メモリをデバイスとして使用する場合には、その外部メモリの接続されているチャンネルでしか再生できません。つまり、デバイス番号0〜1のファイルの<チャンネル番号>は0、デバイス番号2〜3のファイルの<チャンネル番号>は1でなければなりません。 通常は省略することで、音声ファイルを初期化したときのチャンネルが適用されますので、指定する必要はありません。

各パラメータは、音声ファイル番号以外は省略できます。省略時の値は、以下のとおりです。

<rep>=0

<オフセット>=0

<長さ>=ファイルの終わりまで

<サンプリング周波数>=SET PCM文で設定さた値

<チャンネル番号>=SET PCM文で設定された値

CALL REC PCM

機　能

ADPCM/PCMにより音声を音声ファイルに録音します。

書　式

CALL REC PCM(<音声ファイル番号>[, <SYNC>][, <オフセット>][, <長さ>][, <サンプリング周波数>][, <チャンネル番号>])

文　例

```
CALL REC PCM(0)
```

解　説

<音声ファイル番号>で指定されたADPCM/PCM音声ファイルの<オフセット>の位置から<長さ>だけ<サンプリング周波数>で録音します。

パラメータ	値	意　味
<音声ファイル番号>	0～15	録音する音声ファイルを指定します。
<SYNC>	0、1	0はシンクロスタートモードで、入力に音が入って来ると録音を始めます。
<チャンネル番号>	0、1	0は第1チャンネルへの録音、1は第2チャンネルへの録音です。ただし、外部メモリをデバイスとして使用する場合には、その外部メモリの接続されているチャンネルにしか録音できません。つまり、デバイス番号0～1のファイルは<チャンネル番号>は0、デバイス番号2～3のファイルは<チャンネル番号>は1でなければなりません。 通常は省略することで音声ファイルを初期化したときのチャンネルが適用されるので、指定する必要はありません。

各パラメータは音声ファイル番号以外は省略できます。省略時の値は、以下のとおりです。

<SYNC>=0（シンクロスタート機能あり）
<オフセット>=0
<長さ>=ファイルの最後まで
<サンプリング周波数>=SET PCM 文で設定された値
<チャンネル番号>=SET PCM 文で設定された値

CALL SAVE PCM

機　能

ADPCM / PCM 音声ファイルをフロッピーディスクへセーブします。

書　式

CALL SAVE PCM(<ファイル名>, <音声ファイル番号>)

文　例

```
CALL SAVE PCM("DEMO2.DAT", 2)
```

解　説

<音声ファイル番号>で指定された音声ファイルを<ファイル名>で指定されたファイル名でフロッピーディスクに保存します。

<ファイル名>は DISK　BASIC のファイルスペックに適合する文字列で、［ドライブ

名":"]＋最大 8 文字のファイル名＋[".\"最大 3 文字の拡張子]です。
ファイル名としてデバイス名 (CON、PRN、LST、AUX、NUL) を指定すると、そのデバイスに出力されますが､出力途中に CTRL ＋ STOP または CTRL ＋ C などで中止したときは、次の PCM 関係のステートメントは正しく実行されません。その場合、次のステートメントの実行の前に CALL STOPM 文を実行してください。

CALL SET PCM

機　能

ADPCM / PCM の音声ファイルを初期設定します。

書　式

CALL SET PCM(<音声ファイル番号>, <デバイス番号>, <モード>, <パラメータ 1>, <パラメータ 2>[, <サンプリング周波数>][, <チャンネル番号>])

文　例

```
CALL SET PCM(0, 0, 0, , 32)
```

解　説

<音声ファイル番号>で指定された音声ファイルの初期設定をします。
<音声ファイル番号>は 0〜15 で、以下の PCM 関係の命令で参照される番号を実際のデータと結びつける役割をします。

CONVA
CONVP
COPY PCM
LOAD PCM
MK PCM
PLAY PCM
REC PCM
SAVE PCM

<デバイス番号>は以下に示すデバイスを音声ファイルの格納場所に指定します。指定されたデバイスの性質により、その他のパラメータの内容は以下のように変わります。
デバイス番号 0 と 1 は第 1 チャンネルの、デバイス番号 2 と 3 は第 2 チャンネルの MSX-AUDIO LSI の外部メモリを指します。

デバイス番号	デバイス名	モード	パラメータ1	パラメータ2
0	外部RAM (1)	0 or 1	−	長さ
1	外部ROM (1)	−	ROM音声ファイル番号	長さ
2	外部RAM (2)	0 or 1	−	長さ
3	外部ROM (2)	−	ROM音声ファイル番号	長さ
4	メインRAM	0 or 1	配列名	−
5	VRAM	0 or 1	アドレス	長さ

ーは省略すること
長さの単位は256バイト

＜モード＞はPCMのモードで、0のときADPCM、1のときPCMモードに設定されます。外部ROMがデバイスとして指定されたときは、＜モード＞はデータとして指定されているので、省略しなければなりません。また、ROM音声ファイル番号はROMの音声ファイルの番号で、0〜29の値です。下記の「外部ROM内音声ファイルの構造」を参照して下さい。
＜サンプリング周波数＞の単位は[Hz]で、範囲は1800〜16000Hzです。初期値は8000Hzです。
＜チャンネル番号＞は録再チャンネルを指定します。0のとき第1チャンネル、1のとき第2チャンネルが指定されます。外部デバイス(デバイス番号0〜3)については、第1チャンネルのデバイスには第1チャンネルのみ、第2チャンネルのデバイスには第2チャンネルのみしか設定できません。省略された場合は、外部デバイスが指定されている場合には、そのデバイスの接続されているチャンネルが、それ以外の場合には0(第1チャンネル）が指定されます。
CALL AUDIO文で初期化されたときには、音声ファイル番号0が第1チャンネルの外部RAM32Kバイト分に割り当てられ、他の音声ファイルは長さが0となっています。つまり、以下のSET PCM文が実行されたのと同じ状態になっています。

```
CALL SET PCM(0, 0, 0,  , 128)
```

外部ROM内音声ファイルの構造

外部ROMに置かれる音声ファイルの構造は表7.55のようになっています。
ファイルの長さは可変長で、必要な数のディレクトリで管理します。ディレクトリの数は最大30で、最後のディレクトリの後にデータ長として0のディレクトリエントリを置いて最後であることを示します。

表 7.55　外部 ROM 内音声ファイルの構造

オフセット	内　容
0	「A」
1	「B」
2〜15	0（予約）
16〜23	ディレクトリエントリ 0
24〜31	ディレクトリエントリ 1
⋮	⋮
240〜247	ディレクトリエントリ 29（最大）
248〜255	0（ディレクトリ終了マーク）
256〜	音声データ

表 7.56　各ディレクトリエントリの構造

オフセット	内　容	
0	0	ADPCM
	128	PCM
1	0（予約）	
2〜3	スタート番地（256 バイト単位）	
4〜5	データ長（256 バイト単位）	
6〜7	サンプリング周波数	

3. インスツルメント関係のステートメント

ミュージックキーボードとFM音源の結合をインスツルメント(楽器)と呼び、CALL AUDIO文でインスツルメントに使うチャンネル数で決められたチャンネルがミュージックキーボード(MK)と結合されます。この結合はバックグラウンド(背景)で処理されるので、MSXをMKにより演奏する楽器としての使い方をBASICでのプログラム、コマンドの実行と独立して行なうことができます。

CALL INMK

機　能

ミュージックキーボードの変化を知らせます。

書　式

CALL INMK[([<変数1>][, [<変数2>][, [<変数3>]]])]

文　例

```
CALL INMK(A)
CALL INMK(A, B, C)
CALL INMK(A, B)
CALL INMK( ,, C)
```

解　説

ミュージックキーボードの変化を知らせます。キーの変化は割り込みにより検出され、キーバッファに格納されます。パラメータを持っている場合には、キーバッファから1つの変化をとり出し、

<変数1>にキーコード番号
<変数2>にキーのON/OFF
<変数3>にキーコード番号に対応するADPCMの周波数

を入れます。キーコード番号は0～127の範囲で、中央Cは60です。
キーバッファが空の場合、(0, 0, 0)を返します。
アーギュメントが無い場合は、キーバッファをクリアします。
キーバッファの大きさは32です。バッファがオーバフローした場合は、「Device I/O error」になります。その際には、バッファはクリアされます。

CALL KEY ON / KEY OFF

機　能

インスツルメントにキーオン、オフを与えます。

書　式

CALL KEY ON(＜キーコード番号＞[, ＜ベロシティ＞])

CALL KEY OFF(＜キーコード番号＞)

文　例

```
CALL KEY ON(60, 3):CALL KEY OFF(59)
```

解　説

インスツルメントにプログラムで、キーオン、オフを与えます。

＜キーコード番号＞は 0～127 の範囲で、中央 C は 60 に対応します。

＜ベロシティ＞は 0～15 の範囲で省略時は 8 を与えます。

これにより、ミュージックキーボードをプログラムによりエミュレートすることができます。ただし、この命令で与えたキーオン、オフは MK 記録の対象とはなりません。

CALL MK PCM

機　能

インスツルメントとして演奏する ADPCM の音のファイル番号を指定します。

書　式

CALL MK PCM(＜音声ファイル番号＞)

CALL MK PCM(OFF)

文　例

```
CALL MK PCM(1)
```

解　説

インスツルメントで ADPCM を使って演奏する音声ファイル番号を指定､解除します。

＜音声ファイル番号＞は 0～15 です。「OFF」を指定すると、音声ファイル番号が解除されます。

音声ファイルは外部デバイス（デバイス番号0〜3）にADPCMで録音されていなければなりません。

CALL MK TEMPO

機 能

ミュージックキーボード演奏記録、再生とメトロノーム機能の速度を設定します。

書 式

CALL MK TEMPO([<テンポ値>][, <パーカッションマップ>])

文 例

```
CALL MK TEMPO(60)
CALL MK TEMPO(60, 1)
CALL MK TEMPO(, 0)
```

解 説

タイマの周期をコントロールしてMK記録、再生機能やメトロノーム機能の動作速度を設定します。

<テンポ値>は25〜360で、CALL AUDIO文で設定する初期値は120です。このとき、タイマ周期は16.7mSで、四分音符はタイマ周期30個に対応します。

<パーカッションマップ>は0〜31で、以下のようにビットマップで指定されたリズム音によりメトロノーム機能を設定します。

なお、CALL AUDIO文でリズム使用を設定しておく必要があります。

bit0	ハイハットシンバル音
bit1	トップシンバル音
bit2	タムタム音
bit3	スネアドラム音
bit4	バスドラム音

AUDIO文による初期設定値は0で、メトロノーム機能は停止しています。

この命令によって次の機能の速度が影響を受けます。

MK APPEND
MK PLAY
MK REC

CALL MK VEL

機　能

インスツルメントにベロシティを設定します。

書　式

CALL MK VEL(<ベロシティ値>)

文　例

```
CALL MK VEL(15)
```

解　説

インスツルメントにベロシティを設定して初期化します。ベロシティはキーボードのタッチ速度で、強度を表わします。これによって、FM音の音量とともに音質も変化します。

MSX-AUDIOのミュージックキーボードでは一定のベロシティしか発生しませんので、その値を与えます。設定できる<ベロシティ値>は0〜15で、初期値は8です。

このステートメントはインスツルメントを初期化しますので、音は1度途切れます。

CALL MK VOICE

機　能

インスツルメントのボイス（音色の種類）を設定します。

書　式

CALL MK VOICE(<パラメータ1>[, <パラメータ2>])

<パラメータ>=@＋単純変数　または

　　　　　　　　配列変数名

文　例

```
CALL MK VOICE(@2)
```

解　説

インスツルメントの音色の種類を設定します。

<パラメータ>として単純変数が設定されたときは音色番号を指定します。音色番号の

範囲は 0〜63 です。単純変数を指定する場合には、変数の前に「@」をつけます。@記号がない場合には、配列がパラメータとして与えられたと解釈し、その配列の内容が音色パラメータとなります。
音色パラメータの詳細については、「表 7.47 音色パラメータデータのフォーマット」を参照して下さい。
パラメータが 2 つのときは、<パラメータ 1>が第 1 チャンネルの、<パラメータ 2>が第 2 チャンネルの音色になります。
2 つの MSX-AUDIO LSI が実装されている場合に、1 つのパラメータしか設定しないときは、<パラメータ 1>が両方のチャンネルに設定されます。

CALL MK VOL

機 能

インスツルメントの音量を設定します。

書 式

CALL MK VOL(<ボリューム値 1>[, <ボリューム値 2>])

文 例

CALL MK VOL(40)

解 説

インスツルメントの音量を設定します。<ボリューム値>の範囲は 0〜63 で、1 ステップあたり+0.75dB 変化します(8 ステップで 6dB)。CALL AUDIO 文による初期化後の値は 63 (最大音量) です。
このステートメントはキーのオン、オフとは関係なく与えられます。
パラメータが 2 つのときは、<ボリューム値 1>が第 1 チャンネルの、<ボリューム値 2>が第 2 チャンネルのボリューム値になります。
2 つの MSX-AUDIO LSI が実装されている場合に、1 つのパラメータしか設定されないときは、<ボリューム値 1>が両方のチャンネルのボリュームを設定します。

4. MK 記録関係のステートメント

以下では、ミュージックキーボードによるインスツルメントの演奏の記録に関するステートメントを説明します。記録は、ミュージックキーボードから行われ、再生はインスツルメントで行います。

メイン RAM のアドレスを直接指定することにより、記録領域として使用する場合にはバックグラウンド（背景）で行なうことができます。

■ MK 記録のフォーマット

MK 記録はキーオンとキーオフの 2 つのイベントの起きた時刻とキーコード番号を表 7.57 のフォーマットで混在して記録します。

表 7.57　MK 記録のフォーマット

キーオン（3 バイト）

オフセット	内　容
0	ディレイバイト（0〜255）
1	bit0〜6　キーコード番号（0〜127）
	bit7　　キーオンの ID（= 1）
2	bit0〜3　ベロシティ（0〜15）

キーオフ（2 バイト）

オフセット	内　容
0	ディレイバイト（0〜255）
1	キーコード番号（1〜125）

ノーオペレーション（2 バイト）

オフセット	内　容
0	ディレイバイト（0〜255）
1	コード番号（0 または 127）

終了マーク（2 バイト）

オフセット	内　容
0	ディレイバイト（0〜255）
1	コード番号（126）

ディレイバイトはMSX-AUDIO LSIのタイマの周期（MK TEMPO文によって設定）を単位にした値で、直前のイベントからの経過時間を値として取ります。ただし、256以上のタイマ値を記録できるように、キーオフのキーコード番号127および0はノーオペレーション（無効果）としています。

また、キーオフのキーコード番号126はMK記録の終了マークの意味を持っています。キーコード番号は0～127（中央Cが60に対応)、ベロシティは0～15の値です。

CALL APPEND MK

機能

MK記録の追加記録を行ないます。

書式

CALL APPEND MK(<配列名>)
CALL APPEND MK(<開始アドレス>，<終了アドレス>)

文例

```
CALL APPEND MK(A)
```

(配列Aはすでに DIM 文で宣言され、CALL REC MK文により一部分記録されていること)

解説

記録領域の中の終了マークをさがし、その場所からMK記録をします。

CALL CONT MK

機能

CALL STOPM 文により停止したMKの記録、再生を再開します。

書式

CALL CONT MK

文例

```
CALL CONT MK
```

解説

CALL STOPM 文により停止した MK の記録、再生を再開します。

CALL MK STAT

機能

MK 記録システムの状態を知らせます。

書式

CALL MK STAT(<変数名>)

文例

```
CALL MK STAT(A):PRINT A
```

解説

MK 記録システムの状態を知らせます。返される<変数>には、以下の意味があります。

ビット	意　味
7	MK から FM 音源への結合（1 で ON）
4	MK から ADPCM への結合（1 で ON）
3	MK 再生（1 で ON）
2	MK 記録（1 で ON）
1	記録モードの bit1
0	記録モードの bit0

CALL PLAY MK

機能

MK 記録を再生します。

書式

CALL PLAY MK(<配列名>)

CALL PLAY MK(<開始アドレス>, <終了アドレス>)

CALL PLAY MK

文 例

CALL PLAY MK(A)
(配列Aはすでに DIM 文で宣言され、REC MK 文によって記録されていること)

解 説

インスツルメントの演奏を再生します。
パラメータが<配列名>のときは、バックグラウンド処理はできません。パラメータが<メモリアドレス>のときは、バックグラウンド処理を行なうことができます(BGM 文の指定による)。バックグラウンド処理の場合、記録と再生は同時に行なうことができ、記録内容を RECMOD の指定によって切り換えることができます。
パラメータがない場合は最後に記録したものを再生します。
再生を停止するには CTRL + STOP を押す(フォアグラウンド処理時)か、CALL STOPM 文を実行します(バックグラウンド処理時)。CALL STOPM 文で停止した MK 再生は、CONT MK 文により再開できます。

CALL REC MK

機 能

インスツルメントの演奏を記録します。

書 式

CALL REC MK(<配列名>)
CALL REC MK(<開始アドレス>, <終了アドレス>)

文 例

```
100 DIM A(500)
110 CALL REC MK(A)
```

解 説

ミュージックキーボード(MK)の演奏を記録します。記録領域として、配列またはメイン RAM を使用することができます。
<配列>を記録領域として指定したときは、バックグラウンドでの記録はできません。
メモリの<開始アドレス>と<終了アドレス>で指定したときは、バックグラウンドで記録できますが(BGM 文の指定による)、その領域は他の目的(プログラムを置くなど)では使用できません。CLEAR 文で演奏記録用の領域を確保して下さい。
ミュージックキーボードの演奏を記録するか、再生を記録するかは CALL RECMOD 文

で指定します。
記録を中止するには、CTRL + STOP を押す(フォアグラウンド処理時)か、CALL STOPM 文で停止します(バックグラウンド処理時)。CALL STOPM 文で停止した MK 記録 (バックグラウンド) は CONT MK 文により再開できます。
記録されたデータをフロッピーディスクにセーブするには、DISK BASIC の COPY 文で、配列に記録したものをフロッピーディスクに転送します。また、直接アドレスを指定した場合には BSAVE 命令を使うことができます。

CALL RECMOD

機　能

MK 記録の記録モードを設定します。

書　式

CALL RECMOD(<記録モード>)

文　例

```
CALL RECMOD(2)
```

解　説

MK 記録の記録モードを設定します。MK 記録の記録と再生を同時にバックグラウンドで行うときに便利な命令です。初期値は 1 です。
<記録モード>は 0〜3 で、以下の意味があります。

0	ミューティング
1	MK 演奏を記録
2	MK 再生を記録
3	MK 演奏と再生を両方とも記録

4.3 拡張 BIOS

4.3.1 はじめに

MSX-AUDIOでは、アプリケーションソフトウェア用のサービスルーチンとして、拡張BIOSとMBIOS (Music BIOS) コールを用意しています。拡張BIOSコールにより、アプリケーションソフトウェアはそのスロットアドレスやアドレスなどの位置を調べ、インタースロットコールなどを使って、ジャンプテーブルを経由して呼び出します。

高速処理を必要とする場合は、あらかじめスロットをイネーブルしておき、直接コールすることもできます。この章では、MSX-AUDIO拡張BIOSを使用するのに必要な、拡張BIOSコールの方法と各BIOSの機能について解説します。

4.3.2 拡張 BIOS の呼び出し

1. ジャンプテーブルアドレスの取得

アプリケーションは、まず以下の拡張BIOSコールにより、MSX-AUDIO拡張BIOSの存在するスロットとジャンプテーブルの先頭アドレスを調べなければなりません。

拡張BIOSの存在するスロットとジャンプテーブルの先頭アドレスは、以下のようにして求めます。

1. RETURN情報エリア用のワークエリア (64バイト) を取る
2. 以下の設定を行い、FFCAH番地をコールする

コール手順

D	デバイス番号 (10) MSX-AUDIO拡張BIOSのデバイス番号は10
E	機能番号 (0)
B	RETURN情報エリアのスロットアドレス スロットアドレスは、システムワークエリアに保存されている
HL	RETURN情報エリアの先頭アドレス

RAMのスロットアドレスは以下のワークエリアに保存されています。このワークエリアはディスクが接続されているシステムで有効です。ディスクが接続されていないシステムでRAMのスロットアドレスを捜す場合は、サンプルプログラムの「RAMSRCH.MAC」や「AUDIO.MAC」を参照して下さい。

表7.58 RAMのスロットアドレス

ページ	ワークエリアのアドレス
0	F341H
1	F342H
2	F343H
3	F344H

戻り値

B　　次のRETURN情報エリアのスロットアドレス
HL　　次のRETURN情報エリアの先頭アドレス

変更レジスタ

F

RETURN情報はアプリケーションが指定した領域に次のように格納されます。

B:HL →	ジャンプテーブルのスロットアドレス
	ジャンプテーブルの先頭アドレス（下位）
	ジャンプテーブルの先頭アドレス（上位）
	リザーブ（常に0）
B:HL returned →	

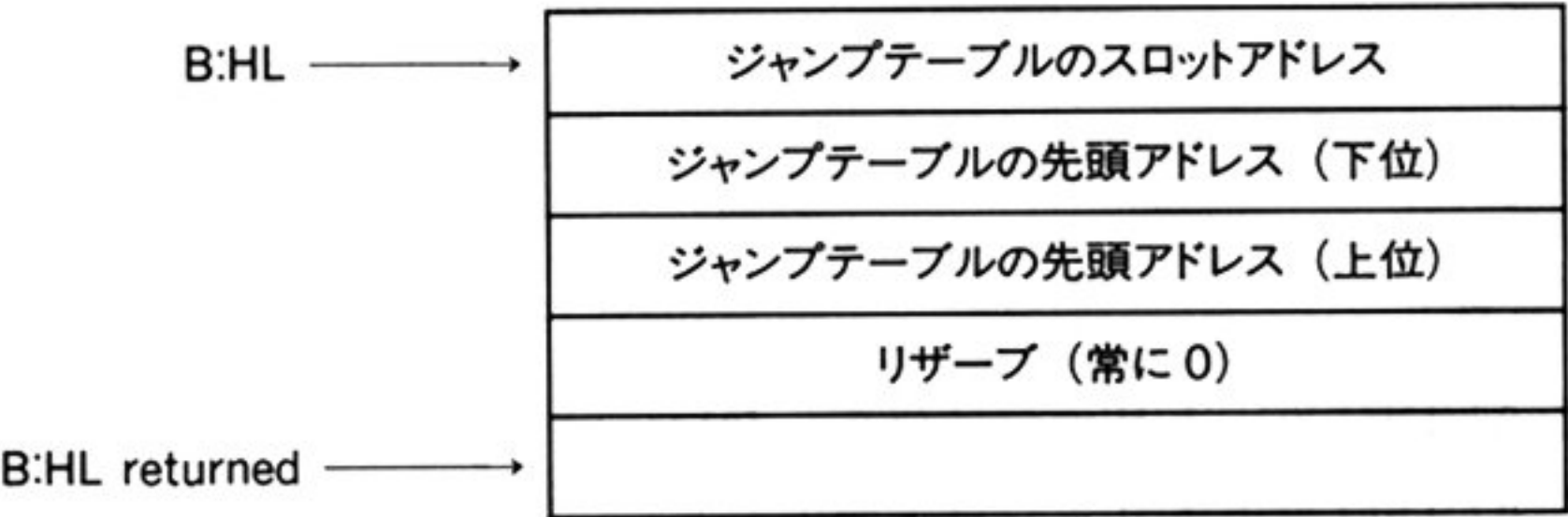



図7.33 RETURN情報の形式

MSX-AUDIOが無いときは、Bレジスタと HL レジスタの内容が変わらずに返ってきます。スロットアドレスの表現は MSX 共通で以下の通りです。

7	6	5	4	3	2	1	0
F	0	0	0	S	S	P	P

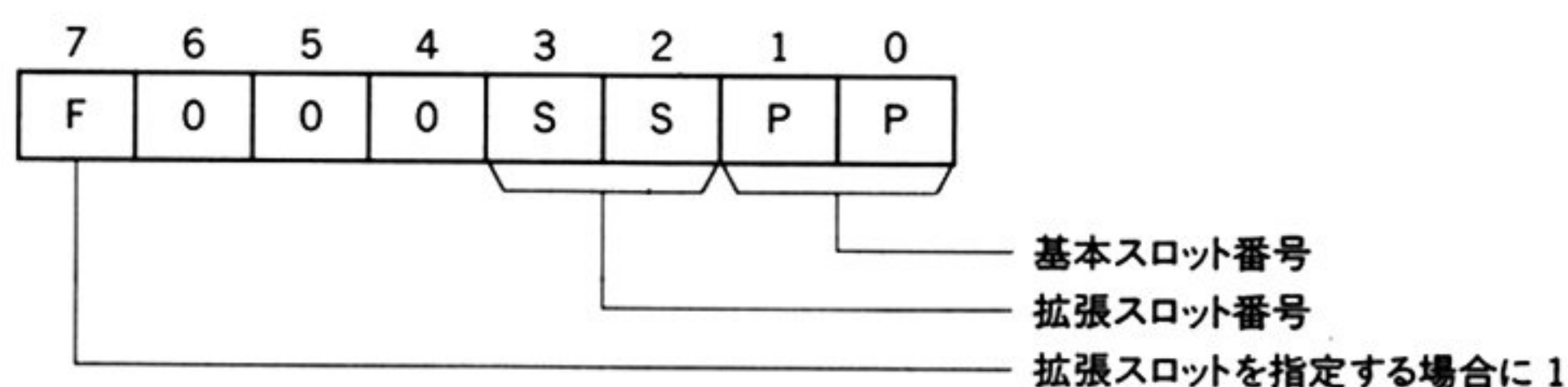



図 7.34 スロットアドレスの形式

拡張 BIOS を使用する場合は、この拡張 BIOS コールで得られたジャンプテーブルをインタースロットコールなどにより呼び出し、目的の BIOS を使用します。

4.3.3 MSX-AUDIO の数のチャンネル獲得

MSX-AUDIO の仕様では、1つの MSX システムに同時に2つの MSX-AUDIO LSI が実装されている場合があります。そのため、MSX-AUDIO が幾つあるかを調べます。

コール手順

D	デバイス番号 (10)
E	機能番号 (1)
A	0

戻り値

A　MSX-AUDIO の数

0	MSX-AUDIO カートリッジは存在しない
1	MSX-AUDIO カートリッジは1つ
2	MSX-AUDIO カートリッジは2つ

変更レジスタ

BC、DE、HL 以外

4.3.4 拡張 BIOS ジャンプテーブルの獲得

MSX-AUDIO 拡張 BIOS は以下に示すジャンプテーブルを持っています。アプリケーションプログラムはインタースロットコールなどで、各エントリを呼び出すことにより、拡張 BIOS の各機能を利用できます。

表 7.59　ジャンプテーブル

先頭からのオフセット(バイト)	名　称	機　能
0	VERSION	ソフトウェアのバージョン番号（現在、3バイトとも0）
3	MBIOS	MBIOS（Music BIOS）の呼び出し
6	AUDIO	MSX-AUDIO の初期化
9	SYNTHE	付属アプリケーションプログラムの呼び出し
12	PLAYF	PLAY 文の動作状態の獲得
15	BGM	BGM モードの設定/解除
18	MKTEMPO	MK（Music Keyboard）再生/記録のテンポの設定
21	PLAYMK	MK 演奏の再生
24	RECMK	MK 演奏の記録
27	STOPM	MK の再生/記録、ADPCM 記録/再生、PLAY 文の停止
30	CONTMK	MK 再生/記録の継続
33	RECMOD	MK 記録モードの設定
36	STPPLY	PLAY 文の停止
39	SETPCM	ADPCM/PCM 領域確保
42	RECPCM	ADPCM/PCM の録音
45	PLAYPCM	ADPCM/PCM の再生
48	PCMFREQ	ADPCM/PCM 再生周波数の変更
51	MKPCM	MK 用 ADPCM データの設定/解除
54	PCMVOL	ADPCM/PCM 再生音量の設定
57	SAVEPCM	ADPCM/PCM データのセーブ
60	LOADPCM	ADPCM/PCM データのロード
63	COPYPCM	ADPCM/PCM データの転送
66	CONVP	ADPCM データを PCM データに変換
69	CONVA	PCM データを ADPCM データに変換
72	VOICE	FM 音源データの設定
75	VOICECOPY	FM 音源データの移動

4.3.5 表記法

拡張BIOSの名称 — AUDIO

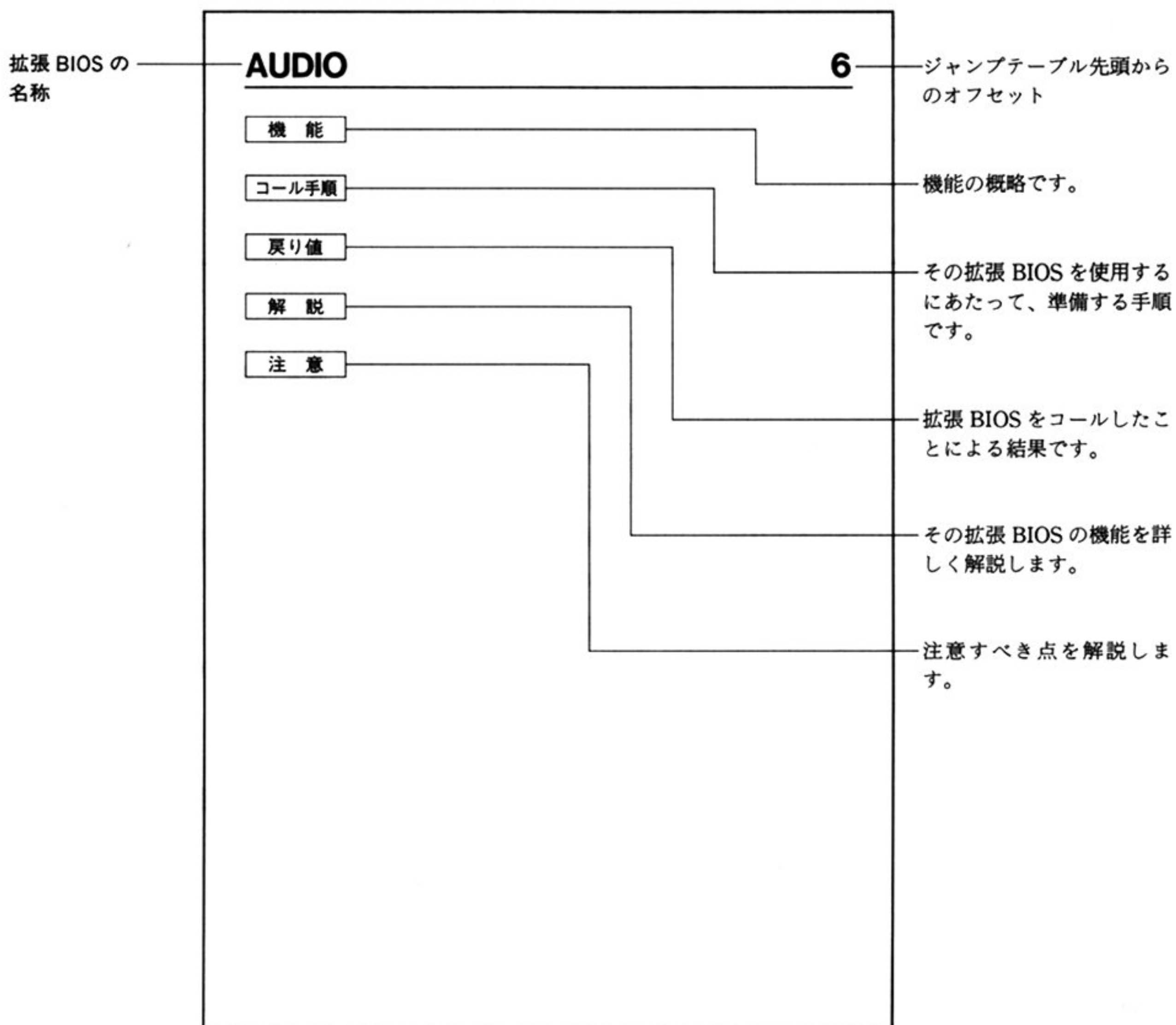
6 — ジャンプテーブル先頭からのオフセット

機能 — 機能の概略です。

コール手順 — その拡張BIOSを使用するにあたって、準備する手順です。

戻り値 — 拡張BIOSをコールしたことによる結果です。

解説 — その拡張BIOSの機能を詳しく解説します。

注意 — 注意すべき点を解説します。

MSX-AUDIOの拡張BIOSコールでは、特に断りがない限り、すべてのレジスタは保存されません。

AUDIO 6

機　能

MSX-AUDIOを初期化します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF	PLAY文で使用するFM音源用文字列の数（0〜9）
+1	モードスイッチ*
+2	インスツルメントに使用するFM音源の数（0〜9）
+3	PLAY文第1文字列で使用するFM音源の数（0〜9）
+4	PLAY文第2文字列で使用するFM音源の数（0〜8）
+5	PLAY文第3文字列で使用するFM音源の数（0〜7）
+6	PLAY文第4文字列で使用するFM音源の数（0〜6）
+7	PLAY文第5文字列で使用するFM音源の数（0〜5）
+8	PLAY文第6文字列で使用するFM音源の数（0〜4）
+9	PLAY文第7文字列で使用するFM音源の数（0〜3）
+10	PLAY文第8文字列で使用するFM音源の数（0〜2）
+11	PLAY文第9文字列で使用するFM音源の数（0〜1）

*モードスイッチ

bit0	0	リズムを使用しない
	1	リズムを使用する
bit1	0	PLAY文でADPCMを扱わない
	1	PLAY文でADPCMを扱う
bit2	0	MSX-AUDIOをCSMモードにしない
	1	MSX-AUDIOをCSMモードにする

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、初期化は行われません。

解　説

MSX-AUDIOシステムを初期化します。「SYNTHE」と「MBIOS」を除くすべての機能は、この初期化をした後、使用できます。

注　意

FM音源の総数はリズムを使用しないときは9まで、使用するときは6までです。

SYNTHE 9

機 能

付属アプリケーションプログラムを呼び出します。

コール手順

なし

戻り値

なし

解 説

付属アプリケーションプログラムに制御を移します。もし、以前にAUDIOが呼ばれていると、なにもせずにリターンします。

注 意

BASICでCLEAR文を実行した後には、付属のアプリケーションプログラムは起動しないで下さい。最悪の場合、暴走する可能性があります。

SETPCM 39

機 能

ADPCM / PCMの音声ファイルを初期設定します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF	音声ファイル番号（0～15）
+1	デバイス番号（0～5、ただし4を除く）
+2	モード（0か1）
+3	デバイス番号によって異なる
+4	デバイス番号によって異なる
+5	長さの下位8ビット
+6	長さの上位8ビット
+7	サンプリング周波数の下位8ビット
+8	サンプリング周波数の上位8ビット

+9　　チャンネル番号（0か1）

BUF+3とBUF+4のパラメータはデバイス番号によって以下のように異なります。

デバイス番号が0か2のとき（外部RAM）
+3　　設定する必要はない
+4　　設定する必要はない

デバイス番号が1か3のとき（外部ROM）
+3　　ROM音声ファイル番号
+4　　必ず0

デバイス番号が5のとき（VRAM）
+3　　VRAMアドレスの下位8ビット
+4　　VRAMアドレスの上位8ビット

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、設定は行われません。

注　意

デバイス番号4（CPU）は使用できません。

RECPCM　42

機　能

音声を音声ファイルに録音します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF　　音声ファイル番号（0〜15）
+1　　SYNC（0か1）
+2　　オフセットの下位8ビット
+3　　オフセットの上位8ビット
+4　　長さの下位8ビット
+5　　長さの上位8ビット
+6　　サンプリング周波数の下位8ビット

+7　　サンプリング周波数の上位8ビット
+8　　チャンネル番号（0か1）

長さ、サンプリング周波数にSETPCMで設定した値を使用するときは、各パラメータに0FFFFHを設定して下さい。
チャンネル番号にSETPCMで設定した値を使用するときは、チャンネル番号に0FFHを設定して下さい。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、録音は行われません。

PLAYPCM　45

機　能

音声ファイルを再生します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF　　音声ファイル番号（0～15）
+1　　REPEAT（0か1）
+2　　オフセットの下位8ビット
+3　　オフセットの上位8ビット
+4　　長さの下位8ビット
+5　　長さの上位8ビット
+6　　サンプリング周波数の下位8ビット
+7　　サンプリング周波数の上位8ビット
+8　　チャンネル番号（0か1）

長さ、サンプリング周波数にSETPCMで設定した値を使用するときは、各パラメータに0FFFFHを設定して下さい。
チャンネル番号にSETPCMで設定した値を使用するときは、チャンネル番号に0FFHを設定して下さい。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、再生は行われません。

PCMFREQ　48

機　能

再生周波数を変更します。

解　説

ローカルモード再生中にサンプリング周波数を変えます。再生中でないと効果はありません。

コール手順

BC　第1チャンネルのサンプリング周波数
DE　第2チャンネルのサンプリング周波数
範囲は1800〜49716で、単位はHzです。
第2チャンネルがないときは、DEレジスタにBCレジスタと同じ値を設定して下さい。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、サンプリング周波数の変更は行われません。

PCMVOL　54

機　能

ADPCM / PCM再生音量を設定します。

コール手順

BC　第1チャンネルの再生音量
DE　第2チャンネルの再生音量
範囲は0〜63で、63が最大音量です。
第2チャンネルが存在しないときは、DEレジスタにBCレジスタと同じ値を設定して下さい。初期値はADPCMでは63、PCMでは32です。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、音量の設定は行われません。

SAVEPCM　57

機　能

ADPCM / PCM 音声ファイルをフロッピーディスクへセーブします。

コール手順

A　　音声ファイル番号

HL　　ファイル名のあるアドレス

ファイル名は DISK BASIC のファイルスペックに適合する文字列の前後にダブルクオート（22H）を置き、最後に 0 を置きます。

```
FILENAME: DB        22H, ''A:VOICE.PCM'', 22H, 0
```

戻り値

音声ファイル番号が正しくないときはキャリーフラグを立てて戻り、セーブは行われません。

注　意

ファイル名が正しくなかったり、ディスクが入っていないなどのエラーが起きると BASIC インタープリタのエラー処理ルーチンに制御が移ります。これを避けるためには、【H.ERRO(FFB1H)】というフックを設定して、ユーザープログラム側でエラー処理を行って下さい。

フロッピーディスク以外のデバイス名を指定しても正しい動作は行われません。

■音声ファイルのフォーマット

音声ファイルは 3 つのブロックからなります。最初は 7 バイトのヘッダブロックです。これは BSAVE ステートメントでセーブされるデータの先頭と同じで、音声ファイルを BLOAD 文でロードできるようになっています。次は 8 バイトのインフォメーションブロックです。ここには、セーブされている ADPCM / PCM データについての情報が書かれています。最後が実際の ADPCM / PCM データです。

ファイルの先頭からのオフセット	ヘッダブロック
0	BSAVE形式と同じID (FEH)
1、2	0000H
3、4	音声データの長さ+インフォメーションブロックの長さ−1 (バイト) ただし音声データの長さが65536バイト以上の場合、FFFFH
5、6	ADPCMデータの場合 0000H PCMデータの場合 0001H

	インフォメーションブロック
7、8	音声データの長さ (256バイト単位)
9、10	サンプリング周波数 (Hz)
11、12	ADPCMデータの場合 初期予測値である8000H PCMデータの場合 0000H
13、14	ADPCMデータの場合 初期量子化幅である007FH PCMデータの場合 0000H

	データブロック
15〜	音声データ

図7.35 音声ファイルのフォーマット

LOADPCM 60

機能

ADPCM/PCM音声ファイルをフロッピーディスクからロードします。

コール手順

A　音声ファイル番号

HL　ファイル名のあるアドレス

ファイル名はDISK BASICのファイルスペックに適合する文字列の前後にダブルクウォート (22H) を置き、最後に0を置きます。

```
FILENAME: DB        22H, ''A:VOICE.DAT'', 22H, 0
```

戻り値

音声ファイル番号が正しくなかったり、SAVEPCMでセーブされたファイルでないファイルをロードしようとしたときは、キャリーフラグを立てて戻り、ロードは行われません。

注　意

ファイル名が正しくなかったり、フロッピーディスクが入っていないなどのエラーが起きるとBASICインタープリタのエラー処理ルーチンに制御が移ります。これを避けるためには【H.ERRO(FFB1H)】というフックを設定してユーザープログラム側でエラー処理を行って下さい。
フロッピーディスク以外のデバイス名を指定しても正しい動作は行われません。

COPYPCM　63

機　能

ADPCM / PCMのデータを音声ファイル間で転送します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF	ソース音声ファイル番号
+1	デスティネーション音声ファイル番号（0〜15）
+2	ソースオフセットの下位8ビット
+3	ソースオフセットの上位8ビット
+4	長さの下位8ビット
+5	長さの上位8ビット
+6	デスティネーションオフセットの下位8ビット
+7	デスティネーションオフセットの上位8ビット
+8	ソース指定（0か1）

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、データの転送は行われません。

CONVP　66

機　能

PCM形式のデータをADPCM形式のデータに変換します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF　ソース音声ファイル番号（0〜15）
+1　デスティネーション音声ファイル番号（0〜15）

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、データの転送は行われません。

CONVA　69

機　能

ADPCM形式のデータをPCM形式のデータに変換します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF　ソース音声ファイル番号（0〜15）
+1　デスティネーション音声ファイル番号（0〜15）

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、データの転送は行われません。

MKTEMPO　18

機　能

ミュージックキーボード演奏記録、再生速度とメトロノーム機能の速度を設定します。

コール手順

DE　　テンポ

範囲は25〜360で、単位は1分間当りの四分音符の数です。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、設定は行われません。

MKPCM　51

機　能

ミュージックキーボードで演奏するADPCM音声ファイルを音声ファイル番号で指定します。

コール手順

A　　音声ファイル番号

範囲は0〜15です。演奏を止めるには0FFHを指定します。

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、指定は行われません。

注　意

ローカルモード以外の音声ファイルを指定した場合、演奏は行われません。

■ミュージックキーボード演奏の記録、再生

ミュージックキーボード（MK）演奏は以下の形式でメモリに記録されます。

キーが押されたときのデータ（3バイト）

オフセット	内　容	
0	ディレイバイト（0〜255）*1	
1	ビット7	キーオンID（1）
	ビット6〜0	キーコード番号（1〜125）*2
2	ベロシティ（0〜15）*3	

キーが離されたときのデータ（2 バイト）

オフセット	内　容	
0	ディレイバイト（0〜255）*1	
1	ビット 7	キーオフ ID（0）
	ビット 6〜0	キーコード番号（1〜125）*2

ノーオペレーション（2 バイト）*4

オフセット	内　容
0	ディレイバイト（0〜255）*1
1	ノーオペレーション ID（0 か 127）

終了マーク　（2 バイト）

オフセット	内　容
0	ディレイバイト（0〜255）*1
1	終了 ID（126）

注　意

*1 ディレイバイト
キーが押されたり離されたりすることをイベントと呼びますが、ディレイバイトとは前のイベントが起きてから今回のイベントが起きるまでの経過時間で、MKTEMPO により MSX-AUDIO LSI に設定されたタイマ割り込みの回数が書かれています。

*2 キーコード
キーコードは音程を指定するコードで中央 C が 60 です。

*3 ベロシティ
ベロシティはキーを押した速度を意味します。現在の MSX-AUDIO では速度検出型の鍵盤をサポートしていないので、あらかじめ設定されたベロシティが書かれます。これは拡張 BASIC では CALL MKVEL で、MBIOS では SM_MK で設定します。

*4 ノーオペレーション
ディレイバイトにより記憶できるイベントの経過時間はタイマ割り込みの回数で 255 までです。よって、それ以上の時間イベントがないときに書かれるデータがノーオペレーションデータです。

PLAYMK 21

機　能

MK 演奏記録を再生します。

コール手順

BC　　MK 演奏記録データの開始番地
DE　　MK 演奏記録データの最終番地

戻り値

なし

RECMK 24

機　能

MK 演奏を記録します。

コール手順

BC　　MK 演奏記録データの開始番地
DE　　MK 演奏記録データの最終番地

戻り値

なし

CONTMK 30

機　能

STOPM によって停止していた MK 再生/記録を継続します。

コール手順

なし

戻り値

なし

RECMOD　33

機　能

MK 記録の記録モードを設定します

コール手順

A　　モード
　　0　　ミューティング（記録しない）
　　1　　MK 演奏を記録
　　2　　MK 再生を記録
　　3　　MK 演奏と再生を両方とも記録

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、設定は行われません。

MBIOS　3

機　能

MBIOS を呼び出します。

コール手順

MBIOS のエントリアドレスを HL レジスタに設定します。また他のレジスタは MBIOS の各ルーチンによって設定が異なります。IX レジスタと IY レジスタはインタースロットコールに使われるため、そのままでは渡すことができません。したがって、【BUF(F55EH)】に以下のように設定します。

BUF	IX レジスタの下位 8 ビット
+1	IX レジスタの上位 8 ビット
+2	IY レジスタの下位 8 ビット
+3	IY レジスタの上位 8 ビット

戻り値

MBIOS の各ルーチンによって異なります。MBIOS については「4.4 MBIOS」をご覧下さい。

PLAYF 12

機　能

PLAY 文の動作状態を調べます。

コール手順

A　PLAY 文のチャンネル番号
範囲は 1～PLAY 文が現在使用できるチャンネル番号です。ただし 0 も指定でき、このときはすべてのチャンネルの指定となります。

戻り値

HL　0　指定されたチャンネルが演奏中ではない
FFFFH　指定されたチャンネルが現在演奏中
すべてのチャンネルが指定されたときは、どれか 1 つのチャンネルでも演奏中ならば 0FFFFH が、そうでなければ 0 が返ります。

BGM 15

機　能

バックグラウンド処理を行うかどうかを指定します。

コール手順

A　0　バックグラウンド処理を行わない
1　バックグラウンド処理を行う
AUDIO を呼び出すと、バックグラウンド処理を行うモードに初期設定されます。

戻り値

なし

注　意

次の機能はバックグラウンド処理ができます。

- PLAY 文による演奏
- ローカルモードの ADPCM 録音再生
- アドレス指定による MK の記録再生

STOPM　27

機　能

MKの記録再生、ADPCM録音再生、PLAY文の演奏などを停止させます。

コール手順

なし

戻り値

なし

解　説

MKの記録再生、ADPCM録音再生、PLAY文の演奏などを停止させます。MKの記録再生はCONTMKで継続することができます。

STPPLY　36

機　能

PLAY文の演奏だけを停止させます。

コール手順

なし

戻り値

なし

VOICE　72

機　能

FM音源の各チャンネルに音色を設定します。

コール手順

【BUF(F55EH)】に以下のパラメータを設定します。

BUF	ボイスパラメータブロック1
+4	ボイスパラメータブロック2
+8	ボイスパラメータブロック3
	⋮
+(n-2)×4	ボイスパラメータブロック n-1
+(n-1)×4	ボイスパラメータブロック n
+n×4	エンドマーク (0FFH)

各ボイスパラメータブロックは4バイトで構成されています。このパラメータは2種類のの指定方法があります。
1つはシステムに備えられている音色データを指定する場合で、以下のようなパラメータブロックで指定します。

+0	FM音源のチャンネル番号 (0〜8)
+1	必ず0を設定
+2	音色ライブラリの音色番号 (0〜63)
+3	必ず0を設定

もう1つはユーザーが用意した音色データを指定する場合で、以下のようなパラメータブロックで指定します。

+0	FM音源のチャンネル番号 (0〜8)
+1	必ず0FFHを設定
+2	音色データのあるアドレスの下位8ビット
+3	音色データのあるアドレスの上位8ビット

戻り値

入力パラメータに誤りがあるとキャリーフラグを立てて戻り、設定は行われません。

解 説

FM音源の各チャンネルに音色を設定します。1度に9チャンネルまでの設定ができます。

VOICECOPY 75

機 能

FM音源データを転送します。

コール手順

【BUF(F55EH)】にパラメータを設定します。

パラメータの指定には以下の方法があります。

1. システム音色ライブラリの0～63のうちの1つをシステム音色ライブラリの32～63のどれかに転送するもので、以下のパラメータを指定します。

BUF	0
+1	ソース音色ライブラリの音色番号（0～63）
+2	0
+3	0
+4	0
+5	0
+6	デスティネーション音色ライブラリの音色番号（32～63）
+7	0
+8	0
+9	0

2. システム音色ライブラリの0～63のうちの1つをユーザーのデータ領域に転送するもので、以下のパラメータを指定します。

BUF	0
+1	ソース音色ライブラリの音色番号（0～63）
+2	0
+3	0
+4	0
+5	0FFH
+6	ユーザーデータ領域のアドレスの下位8ビット
+7	ユーザーデータ領域のアドレスの上位8ビット
+8	0
+9	0

3. ユーザーのデータ領域からシステム音色ライブラリの32～63のどれかに転送するもので、以下のパラメータを指定します。

BUF	0FFH
+1	ユーザーデータ領域のアドレスの下位8ビット
+2	ユーザーデータ領域のアドレスの上位8ビット
+3	0
+4	0
+5	0
+6	デスティネーション音色ライブラリの音色番号（32～63）
+7	0
+8	0

+9　　0

4. システム音色ライブラリの 32〜63 のすべてをユーザーのデータ領域に転送するもので、以下のパラメータを指定します。

BUF	0
+1	0FFH
+2	0
+3	0
+4	0
+5	0FFH
+6	ユーザーデータ領域のアドレスの下位8ビット
+7	ユーザーデータ領域のアドレスの上位8ビット
+8	ユーザーデータ領域の長さの下位8ビット
+9	ユーザーデータ領域の長さの上位8ビット

5. ユーザーのデータ領域からシステム音色ライブラリの 32〜63 のすべてに転送するもので、以下のパラメータを指定します。

BUF	0FFH
+1	ユーザーデータ領域のアドレスの下位8ビット
+2	ユーザーデータ領域のアドレスの上位8ビット
+3	ユーザーデータ領域の長さの下位8ビット
+4	ユーザーデータ領域の長さの上位8ビット
+5	0
+6	0FFH
+7	0
+8	0
+9	0

4.3.6 サンプルプログラム

添付のフロッピーディスクに「AUDIO.MAC」という拡張BIOSコールの実行プログラムが入っています。このプログラムは、以下のBASICプログラムと同等の働きをします。

```
10 PRINT "Go synthesizer (y for yes) ?";
20 A$= INPUT$(1)
30 IF A$="y" OR A$="Y" THEN CALL SYNTHE
40 CALL AUDIO(1,3,1,1,1)
50 END
```

アセンブルはMSX-DOS上で

```
A>M80 = SAMPLE.MAC
```

リンクは

```
A>L80 /P:100,SAMPLE,SAMPLE.BIN/N/E
```

として下さい。

BASICで

```
BLOAD "SAMPLE.BIN",R
```

とすると、プログラムが実行できます。

4.4 MBIOS

4.4.1 はじめに

MBIOS (Music Basic Input Output System) はMSX-AUDIOが持っているFM音源の演奏やADPCM/PCMの録音再生、ミュージックキーボードのスキャンなどをコントロールするプログラム群で、拡張BIOSを通じてアプリケーションプログラムからアクセスすることができます。

サポートする機能には大きく分けて以下のようなものがあります。

- ■ FM音源に対する音色の設定、発声
- ■ リズム音に対する音色の設定、発声
- ■ ADPCM/PCM方式による音声の録音、再生
- ■ ADPCMデータとPCMデータの相互変換
- ■ CSM方式による音声の発声
- ■ ミュージックキーボードのスキャン
- ■ 割り込みのハンドリング

なお、この章を読むにあたっては、「4.5 Y8950」も併読して下さい。

4.4.2 MBIOSのコール手順

MBIOSはY8950を直接アクセスするためのサブルーチン群です。MBIOSをコールするときは、拡張BIOSコールのMBIOS (MBIOSを呼び出すためのエントリ) を使用します。実際の呼び出しは、以下のような手順で行います。

1. MBIOSのエントリアドレスをHLレジスタに設定する。
2. 各MBIOSの指定により、その他のレジスタを設定する。
3. 【BUF(F55EH)】にIX、IYレジスタの内容を設定します。

BUF	IXレジスタの下位8ビット
+1	IXレジスタの上位8ビット
+2	IYレジスタの下位8ビット

+3　　IYレジスタの上位8ビット

4. MSX-AUDIO 拡張 BIOS ジャンプテーブルのオフセット3（MBIOSの呼び出し）をインタースロットコールなどにより呼び出す。

4.4.3 ワークエリア

MBIOS はマスターチャンネル/スレーブチャンネル2つのY8950を制御します。また各々のY8950の9チャンネルのFM音源を制御します。アプリケーションプログラムはMBIOSに対してどのチャンネルを使用するかを、以下の該当するチャンネルのワークエリアのアドレスをMBIOSに渡すことで指定します。

表7.60　チャンネルのワークエリア一覧

ラベル	アドレス	大きさ(バイト)	用　途
CHDB0	3000H	64	FM音源チャンネル0データ
CHDB1	3040H	64	FM音源チャンネル1データ
CHDB2	3080H	64	FM音源チャンネル2データ
CHDB3	30C0H	64	FM音源チャンネル3データ
CHDB4	3100H	64	FM音源チャンネル4データ
CHDB5	3140H	64	FM音源チャンネル5データ
CHDB6	3180H	64	FM音源チャンネル6データ
CHDB7	31C0H	64	FM音源チャンネル7データ
CHDB8	3200H	64	FM音源チャンネル8データ
MIDB_M	3280H	64	ミュージックインスツルメンツデータ（マスターチャンネル）
MIDB_S	32C0H	64	ミュージックインスツルメンツデータ（スレーブチャンネル）
UVL	3700H	1024	ユーザーFM音源データ

例えば、キーオン（RC_KON）は

コール手順

IX　　FM音源チャンネルを指定するためのCHDBのアドレス

となっています。もし、FM音源チャンネル2番に対して指定するなら、

```
rcqkon  equ     35
chdb2   equ     3080h

        ld      a, rcqkon       ; set function code
        ld      ix, chdb2       ; set channel number
        ld      de, 3c00h       ; center 'C'
        ld      c, 8            ; medium velocity
        call    sv_real
```

となります。このキーオンにはマスターチャンネル、スレーブチャンネルの指定がありません。指定がない機能はマスターチャンネル、スレーブチャンネル両方に対して同じ動作をします。

ワークエリアは3000H〜3FFFHの4Kバイトですが、7000H〜7FFFHにもイメージを持ち、どちらのアドレスでも同じように指定できます。

1. MIDB

MIDB（Music Instrument Data Block）はマスターチャンネル、スレーブチャンネルそれぞれに用意されており（MIDB-M（3280H）とMIDB-S（32C0H））、アプリケーションソフトウェアはMBIOSを呼び出すときに、どちらのチャンネルに対する機能要求であるのかを示すために該当するチャンネルのMIDBのアドレスをMBIOSに渡します。

Y8950が持っている各レジスタは書き込みのみで、設定した内容を読み出すことができないため、MBIOSはレジスタに書き込んだ値をMIDBにコピーして保存しています。アプリケーションプログラムはMIDBの内容を参照することができますが、内容を変更してはいけません。

MIDBの内容は以下のとおりです。MIDBの詳細な内容については、「4.4.5 ワークエリアの詳細」を参照して下さい。

表 7.61　MIDB 内容一覧

オフセット	名　称
0	未使用
1	未使用
2	YM_TIM1
3	YM_TIM2
4～17	未使用
18	YMA_BIAS
19～24	未使用
25	YMA_AUDIO
26～31	未使用
32	YMA_TRANS（下位8ビット）
33	YMA_TRANS（上位8ビット）
34	YMA_LFO
35	YMA_RAM
36	ZMA_FLAG
37	ZMA_PDB（下位8ビット）
38	ZMA_PDB（上位8ビット）
39	ZMA_PH_FILTER
40	ZMA_PH_TL
41	ZMA_PH_AR（下位8ビット）
42	ZMA_PH_AR（上位8ビット）
43	ZMA_PH_D1R（下位8ビット）
44	ZMA_PH_D1R（上位8ビット）
45	ZMA_PH_SL
46	ZMA_PH_D2R（下位8ビット）
47	ZMA_PH_D2R（上位8ビット）
48	ZMA_PH_RR（下位8ビット）
49	ZMA_PH_RR（上位8ビット）
50	ZMA_PH_EG（下位8ビット）
51	ZMA_PH_EG（上位8ビット）
52	ZMA_PH_STAT
53～63	未使用

2. CHDB

CHDB（CHannel Data Block）はFM音源の9つの各チャンネルごとに用意されており、MBIOSにより各音源チャンネルの制御のために使用します。

アプリケーションソフトウェアはMBIOSを呼び出すときに、どのチャンネルに対する機能要求であるかを示すために該当するチャンネルのCHDBのアドレスを指定します。

Y8950が持っている各レジスタは書き込みのみで、設定した内容を読み出すことができないの

で、MBIOSはレジスタに書き込んだ値をCHDBにコピーして保存しています。アプリケーションプログラムはCHDBの内容を参照することができますが、内容を変更してはいけません。

CHDBの内容は以下のとおりです。CHDBの詳細な内容については、「4.4.5 ワークエリアの詳細」を参照して下さい。

表7.62 CHDB内容一覧

オフセット	名 称
0	YCAO0_MULTI
1	YCAO0_LS
2	YCAO0_AR
3	YCAO0_RR
4	YCAO0_VELS
5	YCAO0_VTL
6	未使用
7	未使用
8	YCAO1_MULTI
9	YCAO1_LS
10	YCAO1_AR
11	YCAO1_RR
12	YCAO1_VELS
13	YCAO1_VTL
14	未使用
15	未使用
16	YCA_VTRANS（下位8ビット）
17	YCA_VTRANS（上位8ビット）
18	YCA_TRANS（下位8ビット）
19	YCA_TRANS（上位8ビット）
20	YCA_TRIG
21	YCA_VOL
22	YCA_FB
23	YCA_VEL
24	YCA_PITCH（下位8ビット）
25	YCA_PITCH（上位8ビット）
26	YCA_VOICE
27	ZCA_FLAG
28	ZC_CH
29	ZC_OP
30	ZC_COUNT（下位8ビット）
31	ZC_COUNT（上位8ビット）

4.4.4 各エントリの解説

1. MBIOSの初期化

SV_RESET (0090H)

機　能

MBIOSを初期化します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

解　説

Y8950とMBIOS全体を初期状態に設定します。MBIOSのほとんどの機能はこのファンクションを実行した後、SM_AUDIOをコールすると使用できるようになります。

注　意

この機能を呼んだ後は、割り込みは禁止状態（DI）になっています。割り込みを許可する前には、MBIOSの各フックエリアをアプリケーションが設定しなければなりません(4.4.4 9. 割り込みサービス　割り込み処理用ワークエリア参照)。

2. ユーザー割り込みサービスの禁止

SV_DI (0093H)

機　能

ユーザー割り込みを禁止します。

コール手順

なし

戻り値

なし

変更レジスタ

なし

解　説

ユーザー割り込み許可カウンタを1増やします。このカウンタが0のときに限り、ユーザー割り込みの処理が行われます。

注　意

このカウンタは8ビットなので、255回以上連続しての呼び出しはできません。ユーザー割り込みに関しては、「4.4.4 9. 割り込みサービス　MBIOSの割り込み処理」を参照して下さい。

3. ユーザー割り込みサービスの許可

SV_EI (0096H)

機　能

ユーザー割り込みを許可します。

コール手順

なし

戻り値

なし

変更レジスタ

なし

解　説

ユーザー割り込み許可カウンタが0でなければ1減らします。このカウンタが0のとき

に限りユーザー割り込みの処理が行われます。

注 意

ユーザー割り込みに関しては、「4.4.4. 9. 割り込みサービス　MBIOSの割り込み処理」を参照して下さい。

4. Y8950への書き込み（割り込み許可状態）

SV_ADW（0099H）

機 能

Y8950のレジスタにデータを書き込みます。

コール手順

IY　マスター、スレーブのどちらに対する操作かをMIDBのアドレスで指定します。

A　書き込むデータ

C　レジスタ番号

戻り値

CYフラグ　存在しないスレーブチャンネルのY8950に対して書き込みしようとした場合は1にセットされます。

変更レジスタ

IY以外のすべてのレジスタ

注 意

処理の間、割り込みは禁止され、処理の後に許可されます

5. Y8950への書き込み（割り込み禁止状態）

SV_ADW_DI（009CH）

機 能

Y8950のレジスタにデータを書き込みます。

コール手順

IY　マスター、スレーブのどちらに対する操作かをMIDBのアドレスで指定します。
A　書き込むデータ
C　レジスタ番号

戻り値

CYフラグ　存在しないスレーブチャンネルのY8950に対して書き込みしようとした場合は1にセットされます。

変更レジスタ

IY以外のすべてのレジスタ

注　意

処理の間、割り込みは禁止され、処理の後も禁止のままです。

6. 諸機能の初期設定

SV_SETUP（00ABH）

機　能

諸機能の初期設定

コール手順

A　機能コード
　その他のパラメータは機能によって異なります。

戻り値

非同期割り込み機能によって呼び出されたプログラムやユーザー割り込み機能によって呼び出されたプログラムから呼び出された場合は初期設定は行われずキャリーフラグが1になります。

変更レジスタ

IX、IY以外のレジスタ（他のレジスタは各機能により異なる）。

解　説

MBIOSの各機能を初期設定します。この機能はコード指定によりいくつかの機能にわかれています。

SV_SETUPには表7.63の機能があります。以下では、その機能について説明します。なお、コードとはSV_SETUPのAレジスタに入れる機能コードを、変更レジスタとはIX、IY以外の壊されるレジスタを意味します。

表7.63　SV_SETUPの機能一覧

ラベル	コード	機　能
SM_AUDIO	0	楽音機能の設定
SC_CHDB	1	CHDBワークエリアの初期化
SM_INST	2	インスツルメント機能の初期化
SM_MK	3	ミュージックキーボードスキャナーの初期化

SM_AUDIO

コード

0

機　能

楽音機能を設定します。モードは以下の中から選択し設定します。

- FM9音同時発声モード
- FM6音、リズム5音モード
- CSMモード

またCSMモード以外のときは、どのFM音源チャンネルをインスツルメントに割り当てるかを指定します。

コール手順

C　目的によって、以下の設定をします。

ビット7〜3	0		
ビット2	リズムモードの指定		
	0	FM9音同時発声モード	
	1	FM6音、リズム5音モード	
ビット1	マスターチャンネルCSMモード		
	0	FM音源モード	
	1	CSMモード	

ビット0	スレーブチャンネル CSM モード	
	0	FM 音源モード
	1	CSM モード

FM 音源モードを指定する場合、FM 音源のチャンネル番号に相当する DE レジスタのビットを1にすることにより、その FM 音源チャンネルをインスツルメントに割り当てます。
例えば、DE レジスタに 0101B を入れると、FM 音源チャンネル0番と2番がインスツルメントに使われます。リズム音を使用するときは、FM 音源チャンネル6番~8番がリズム専用になるので、他の用途には使用できません。

戻り値

なし

変更レジスタ

すべて

注　意

この機能は内部で SC_CHDB と SM_INST を呼び出しています。

SC_CHDB

コード

1

機　能

CHDB ワークエリアを初期化します。音色データは音色番号0番のデータに初期化されます。

コール手順

IX　　初期化する CHDB のアドレス

戻り値

なし

変更レジスタ

すべて

SM_INST

コード

2

機　能

インスツルメント機能を初期化します。インスツルメントの音色は音色番号0番になります。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

SM_MK

コード

3

機　能

ミュージックキーボードスキャナーを初期化します。

コール手順

B	1	ミュージックキーボードをインスツルメントに接続する
	0	ミュージックキーボードをインスツルメントに接続しない
C	0〜15	ミュージックキーボードが押されたときのベロシティ範囲 0が最弱で15が最強です。このベロシティはSV_MK（ミュージックキーボードのスキャン）で参照されます。

戻り値

なし

変更レジスタ

すべて

7. リアルタイムオペレーション

SV_REAL (00AEH)

機　能

リアルタイムオペレーション（実際の音声の発声など）を行います。この機能はコード指定によりいくつかの機能にわかれています。

コール手順

A　　機能コード
　　その他のパラメータは機能によって異なります。

戻り値

CY フラグ　正しくない入力パラメータで呼ばれた場合は1にセットされます。

変更レジスタ

IX、IY 以外のレジスタ（他のレジスタは各機能により異なる）。

注　意

SV_REAL が実行されている間、UISV は禁止され（DI 状態になっているわけではありません）、戻るときに許可されます。
SV_REAL には表7.64〜表7.68の機能があります。以下で、その機能について説明します。なお、コードとは SV_REAL の A レジスタに入れる機能コードを、変更レジスタとは IX、IY 以外の壊されるレジスタを意味します。

表7.64　SV_SETUP の機能一覧（チャンネルオペレーション）

ラベル	コード	機　能
RC_NOTE	32	指定された FM 音源チャンネルのキーオン、キーオフ
RC_LEGATO	33	指定された FM 音源チャンネルのレガートオン、キーオフ
RC_DAMP	34	発声中の FM 音源チャンネルを強制的に停止
RC_KON	35	指定された FM 音源チャンネルをキーオン
RC_LEGATO_ON	36	指定された FM 音源チャンネルをレガートオン
RC_KOFF	37	発声中の FM 音源チャンネルをキーオフ
RCA_PARAM	38	FM 音源チャンネルへのリアルタイムパラメータの設定
RCA_VOICE	39	FM 音源チャンネルへのボイスの設定（音色番号指定）
RCA_VPARAM	40	FM 音源チャンネルへのボイスパラメータの設定
RCA_VOICEP	41	FM 音源チャンネルへのボイスの設定（音色データ指定）

表 7.65　SV_SETUP の機能一覧（マスターインスツルメント）

ラベル	コード	機　能
RM_TIMER	16	タイマ割り込みの禁止・許可
RM_TIM1	17	タイマ 1 の周期の設定
RM_TIM2	18	タイマ 2 の周期の設定
RM_TEMPO	19	タイマ 2 の周期をテンポで設定
RM_DAMP	20	発声中のすべての FM 音源チャンネルを強制的に停止
RM_PVEL	44	各リズム音のベロシティを設定
RM_PERC	21	リズム音を発声
RMA_MK	22	ミュージックキーボードをスキャンし、その状態を返す
RMA_LFO	23	振幅変調やビブラートの深さの設定
RMA_TRANS	24	現在発声中の音および以降の音のトランスポーズの設定
RM_UTEMPR	28	平均律の音程を設定されている音律の音程への変換
RM_CTEMPR	29	音律の設定
RM_PITCH	30	現在発声中の音および以降の音のピッチの設定
RM_TSRAN	31	現在発声中の音および以降の音のトランスポーズの設定

表 7.66　SV_SETUP の機能一覧（インスツルメント）

ラベル	コード	機　能
RI_DAMP	48	割り当てられた FM 音源チャンネルすべての強制的な停止
RI_ALLOFF	49	割り当てられた FM 音源チャンネルすべてのキーオフ
RI_EVENT	50	指定された音程を音律の変換をしてキーオン、キーオフ
RI_PCHB	51	ピッチベンダーの位置の設定
RI_PCHBR	52	ピッチベンダーが音程に与える度合の設定
RIA_PARAM	53	FM 音源チャンネルへのリアルタイムパラメータの設定
RIA_VOICE	54	FM 音源チャンネルへの音色番号でのボイスの設定
RIA_VPARAM	55	FM 音源チャンネルへのボイスパラメータの設定
RIA_VOICEP	56	FM 音源チャンネルへの音色データでのボイスの設定

表7.67 SV_SETUP の機能一覧 (ADPCM / PCM)

ラベル	コード	機 能
RM_MOVE_DI	0	各デバイス間の ADPCM / PCM データの転送
RM_READ_DI	26	デバイスの ADPCM / PCM データのメイン RAM への256バイト転送
RM_WRITE_DI	27	メイン RAM の 256 バイトの ADPCM / PCM データのデバイスへの転送
RM_TRACE_DI	1	初期の予測値と量子化幅を元にした ADPCM データのトレース
RM_CONV_PCM_DI	2	初期の予測値と量子化幅を元にした ADPCM データのPCM のデータへの変換
RM_CONV_ADPCM_DI	3	初期の予測値と量子化幅を元にした PCM のデータのADPCM データへの変換
RMA_DAC_BIAS	4	PCM 再生を行うときの音量(Y8950 のレジスタ 17H)の設定
RMA_DAC_DI	5	PCM データの再生
RMA_ADC_DI	6	PCM データの録音
RMA_ADPCM_BIAS	7	ADPCM 再生を行うときの音量の設定
RMA_ADPLAY_DI	8	非ローカルモードでの ADPCM データの再生
RMA_ADREC_DI	9	非ローカルモードでの ADPCM データの録音
RMA_ADPLY_SAMPLE	43	ローカルモード再生中のサンプリング周波数の変更
RMA_BREAK	10	ローカルモードでの再生・録音の中断
RMA_ADPLAY	11	ローカルモードでの ADPCM データの再生
RMA_ADREC	12	ローカルモードでの ADPCM データの録音
RMA_ADPLAYLP	42	ローカルモードでの ADPCM データの再生(繰り返し)
RMA_PHASE_SET_DI	13	メインRAM内の256バイトのPCMデータのADPCMデータへの変換
RMA_PHASE_EG	14	エンベロープデータの設定
RMA_PHASE_EVENT	15	指定された音程のサンプリングキーボードシミュレーションでのキーオン、キーオフ

表7.68 SV_SETUP の機能一覧 (CSM 再生)

ラベル	コード	機 能
RMA_CSM_DI	25	CSM データの再生

■チャンネルオペレーション

チャンネルオペレーションは各FM音源チャンネルに対して処理を行います。IX レジスタでCHDB のアドレスを渡すことにより、どの FM 音源チャンネルに対する処理かを指定します。また、チャンネルオペレーションはマスターチャンネルとスレーブチャンネルの両方に対して同じ処理をします。

RC_NOTE

コード

32

機　能

指定された FM 音源チャンネルをキーオンし、指定時間経過後自動的にキーオフします。

コール手順

IX　FM 音源チャンネルを指定するための CHDB のアドレス
DE　音程（0〜32767）
　　中央 C が 15360（3C00H）で、256 の変化で半音変化します。
C　キーを押す速さ（0〜15）
　　0 が最弱、15 が最強です。
B　キーオンの時間（1〜255）
　　SV_TEMPO（自動キーオフ処理）がこの回数呼ばれるとキーオフします。

戻り値

なし

変更レジスタ

すべて

RC_LEGATO

コード

33

機　能

指定された FM 音源チャンネルをレガートオンし、指定時間経過後自動的にキーオフします。

コール手順

IX　FM 音源チャンネルを指定するための CHDB のアドレス
DE　音程（0〜32767）
　　中央 C が 15360（3C00H）で、256 の変化で半音変化します。

B　　キーオンの時間（1〜255）
SV_TEMPO がこの回数呼ばれるとキーオフします。

戻り値

なし

変更レジスタ

すべて

注　意

レガートオンはキーオンと違い、エンベロープの開始を行わないので、1度キーオンした音の音程や強さを変化させるために使用します。

RC_DAMP

コード

34

機　能

発声中のFM音源チャンネルを強制的に停止します。

コール手順

IX　　FM音源チャンネルを指定するためのCHDBのアドレス

戻り値

なし

変更レジスタ

すべて

RC_KON

コード

35

機　能

指定されたFM音源チャンネルをキーオンします。

コール手順

IX　FM音源チャンネルを指定するためのCHDBのアドレス

DE　音程（0〜32767）
中央Cが15360（3C00H）で、256の変化で半音変化します。

C　キーを押す速さ（0〜15）
0が最弱、15が最強です。

戻り値

なし

変更レジスタ

すべて

RC_LEGATO_ON

コード

36

機　能

指定されたFM音源チャンネルをレガートオンします。

コール手順

IX　FM音源チャンネルを指定するためのCHDBのアドレス

DE　音程（0〜32767）
中央Cが15360（3C00H）、256の変化で半音変化します。

戻り値

なし

変更レジスタ

すべて

注　意

レガートオンはキーオンと違い、エンベロープの開始を行わないので、1度キーオンした音の音程や強さを変化させるために使用します。

RC_KOFF

コード

37

機　能

発声中のFM音源チャンネルをキーオフします。

コール手順

IX　　FM音源チャンネルを指定するためのCHDBのアドレス

戻り値

なし

変更レジスタ

すべて

RCA_PARAM

コード

38

機　能

FM音源チャンネルにリアルタイムパラメータを設定します。

コール手順

IX　　FM音源チャンネルを指定するためのCHDBのアドレス
C　　リアルタイムパラメータのCHDB先頭からのオフセット
DE　　設定データ

この機能で設定できるデータは、以下のとおりです。

YCA_TRANS
YCA_VOL
YCA_TRIG
YCA_VEL
YCA_PITCH

戻り値

なし

変更レジスタ

すべて

RCA_VOICE

コード

39

機　能

FM音源チャンネルに音色番号でボイスを設定します。

コール手順

IX	FM音源チャンネルを指定するためのCHDBのアドレス
C	音色番号（0～63）

戻り値

なし

変更レジスタ

すべて

RCA_VPARAM

コード

40

機　能

FM音源チャンネルにボイスパラメータを設定します。

コール手順

IX	FM音源チャンネルを指定するためのCHDBのアドレス
C	リアルタイムパラメータのCHDB先頭からのオフセット
DE	設定データ

この機能で設定できるパラメータは、以下のとおりです。

YCAO0_MULTI、YCAO1_MULTI
YCAO0_LS、YCAO1_LS
YCAO0_AR、YCAO1_AR
YCAO0_RR、YCAO1_RR
YCAO0_VELS、YCAO1_VELS
YCAO0_VTL、YCAO1_VTL
YCA_VTRANS
YCA_FB

戻り値

なし

変更レジスタ

すべて

RCA_VOICEP

コード

41

機　能

FM 音源チャンネルに音色データでボイスを設定します。

コール手順

IX	FM 音源チャンネルを指定するための CHDB のアドレス
BC	音色データのアドレス

戻り値

なし

変更レジスタ

すべて

FM 音源（オペレータ）データの構造

FM 音源の音色を設定するときに使用するデータは、32 バイトで構成されています。

最初の8バイトはその音色につけられた名前で、次の8バイトは2つのオペレータに共通のデータです。残りの16バイトは各オペレータ用に8バイトずつにわかれています。このデータは音色設定時にCHDBにコピーされます。

表7.69 FM音源データ内容一覧

オフセット	ラベル	意味
0〜7	V_NAME	音色の名称を8バイトの文字列で設定
8	V_TRANS(下位8ビット)	音程を計算するときのトランスポーズ値の設定
9	V_TRANS(上位8ビット)	
10	V_ARG	ビット7　振幅変調の深さ 0　1dB 1　4.8dB ビット6　ビブラートの深さ 0　7セント 1　14セント ビット5　振幅変調/ビブラート深さ Y8950では各チャンネルごとに深さを設定することができません。そのため、どれか音色を設定したときに深さを設定できます。このビットはビット6、7の深さをY8950に対して設定することを指定します。 0　設定しない 1　設定する ビット4　固定音程音色 このビットはY8950にはなく、MBIOSで実現している固定音程音色であることを指定します。 0　通常の音色 1　固定音程音 ビット3〜1　フィードバック量の指定 0〜7 ビット0　コネクション 2つのオペレータの結合を指定します。 0　直列周波数変調モード 1　並列サイン波合成モード
11〜15	未使用	
16	VO0_MULTI(オペレータ0)	Y8950のレジスタ番号20H(チャンネル0)〜35H(チャンネル8)に設定されるデータを指定 ビット7　AM(振幅変調指定) 0　振幅変調なし 1　振幅変調あり ビット6　VIB(ビブラート指定) 0　ビブラートなし 1　ビブラートあり

オフセット	ラベル	意　味
		ビット5　　EG-TYP(エンベロープタイプ) 　0　　減衰音 　1　　持続音 ビット4　　KSR（キースケールレート） 　0　　キースケールレートなし 　1　　キースケールレートあり ビット3〜0　　MULTIPLE（マルチプル） 実際に発生される周波数とB-number、F-numberとの倍率を指定します。この値が大きくなれば、同じB-number、F-numberでも実際に発生される周波数は高くなります。 0　　0.5倍 1〜10　　1倍〜10倍 11　　10倍 12　　12倍 13　　12倍 14　　15倍 15　　15倍
17	VO0_TL(オペレータ0)	Y8950のレジスタ番号40H(チャンネル0)〜55H(チャンネル8)に設定するデータを指定 ビット7と6　　KSL（レベルキースケール） ビット5〜0　　トータルレベル
18	VO0_AR(オペレータ0)	Y8950のレジスタ番号60H(チャンネル0)〜75H(チャンネル8)に設定するデータを指定 ビット7〜4　　AR（アタックレート） ビット3〜0　　DR（ディケイレート）
19	VO0_RR(オペレータ0)	Y8950のレジスタ番号80H(チャンネル0)〜95H(チャンネル8)に設定するデータを指定 ビット7〜4　　SL（サスティンレベル） ビット3〜0　　RR（リリースレート）
20	VO0_VELS(オペレータ0)	MBIOSではY8950が持っていない音の強弱をソフトウェアで実現していますが、その強弱の指定に対する実際の強弱の変化の度合を設定します。この値が大きければ、強弱の指定がより有効になります。小さければ、実際の強弱の変化が少なくなり、0の場合は強弱の指定は無効になります。 ビット7〜4　　無効 ビット3〜0　　ベロシティセンシティビティ
21〜23	未使用	
24	VO1_MULTI(オペレータ1)	オペレータ0と同じ
25	VO1_TL(オペレータ1)	〃
26	VO1_AR(オペレータ1)	〃
27	VO1_RR(オペレータ1)	〃
28	VO1_VELS(オペレータ1)	〃
29〜31	未使用	

音色一覧

略号は音色名として音源データに登録されているものです。

表 7.70 システム音色ライブラリ (SVL) 一覧

音色番号	音色名	略　号
0	Piano 1	Piano 1
1	Piano 2	Piano 2
2	Violin	Violin
3	Flute 1	Flute
4	Clarinet	Clarinet
5	Oboe	Oboe
6	Trumpet	Trumpet
7	Pipe Organ 1	PipeOrgn
8	Xylophone	Xylophon
9	Organ	Organ
10	Guitar	Guitar
11	Santool 1	Santool
12	Electric Piano 1	Elecpian
13	Clavicode 1	Clavicod
14	Harpsicode 1	Harpsicd
15	Harpsicode 2	Harpscd2
16	Vibraphone	Vibraphn
17	Koto	Koto
18	Taiko	Taiko
19	Engine 1	Engine
20	UFO	UFO
21	Synthesizer Bell	SynBell
22	Chime	Chime
23	Synthesizer Bass	SynBass
24	Synthesizer	Synthsiz
25	Synthesizer Percussion	SynPercu
26	Synthesizer Rhythm	SynRhyth
27	Harm Drum	HarmDrum
28	Cowbell	Cowbell
29	Close Hi-hat	ClseHiht
30	Snare Drum	SnareDrm
31	Bass Drum	BassDrum

表7.71 ユーザー音色ライブラリ（UVL）一覧

音色番号	音色名	略 号
32	Piano 3	Piano 3
33	Electric Piano 2	Elecpia2
34	Santool 2	Santool2
35	Brass	Brass
36	Flute 2	Flute 2
37	Clavicode 2	Clavicd2
38	Clavicode 3	Clavicd3
39	Koto 2	Koto 2
40	Pipe Organ 2	PipeOrg2
41	PohdsPLA	PohdsPLA
42	RohdsPRA	RohdsPRA
43	Orch L	Orch L
44	Orch R	Orch R
45	Synthesizer Violin	SynViol
46	Synthesizer Organ	SynOrgan
47	Synthesizer Brass	SynBrass
48	Tube	Tube
49	Shamisen	Shamisen
50	Magical	Magical
51	Huwawa	Huwawa
52	Wander Flat	WnderFlt
53	Hard Rock	Hardrock
54	Machine	Machine
55	Machine V	MachineV
56	Comic	Comic
57	SE-Comic	SEQComic
58	SE-Laser	SEQLaser
59	SE-Noise	SEQNoise
60	SE-Star 1	SEQStar
61	SE-Star 2	SEQStar2
62	Engine 2	Engine 2
63	Silence	Silence

■マスターインスツルメント

マスターインスツメントはマスターチャンネルまたはスレーブチャンネルのどちらかに対して処理を行います。IY レジスタで MIDB のアドレスを渡すことにより、マスターチャンネル、スレーブチャンネルのどちらかに対する処理かを指定します。ただし、タイマ関連の処理と RM_DAMP、および、リズム関連の処理はマスターチャンネル、スレーブチャンネル両方に対して処理を行います。

RM_TIMER

コード

16

機　能

タイマ割り込みを禁止・許可します。

コール手順

C	モード		
	ビット 7〜2	無効	
	ビット 1	タイマ 2 の禁止・許可	
		0	禁止
		1	許可
	ビット 0	タイマ 1 の禁止・許可	
		0	禁止
		1	許可

戻り値

なし

変更レジスタ

すべて

RM_TIM1

コード

17

機 能

タイマ1の周期を設定します。

コール手順

C　　周期

分解能80μSのタイマです。0で20.48mS、255で80μSです。

戻り値

なし

変更レジスタ

すべて

RM_TIM2

コード

18

機 能

タイマ2の周期を設定します。

コール手順

C　　周期

分解能320μSのタイマです。0で81.92mS、255で320μSです。

戻り値

なし

変更レジスタ

すべて

RM_TEMPO

コード

19

機　能

タイマ 2 の周期をテンポで設定します。

コール手順

C　　テンポ

1 分間の四分音符の数で指定します。

戻り値

なし

変更レジスタ

すべて

RM_DAMP

コード

20

機　能

発声中のすべての FM 音源チャンネルを強制的に停止します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

注　意

RC_DAMP を使えば個別のチャンネルをダンプできますが、RM_DAMP では全部のチャンネルを同時にダンプします。

RM_PVEL

コード

44

機　能

各リズム音のベロシティを設定します。

コール手順

C　　リズムの種類指定
設定する以下のリズム楽器のビットを1にします。同時に2つ以上の設定ができます。

ビット7〜5	無効
ビット4	バスドラム
ビット3	スネアドラム
ビット2	タムタム
ビット1	トップシンバル
ビット0	ハイハット

E　　ベロシティ（0〜31）
0が最強で、31が最弱です。

戻り値

なし

変更レジスタ

すべて

注　意

選択されなかったリズム音のベロシティは変化しません。

RM_PERC

コード

21

機　能

リズム音を発声します。

コール手順

C　　リズムの種類指定

発声する以下のリズム楽器のビットを1にします。同時に2つ以上の発声ができます。

ビット7〜5	無効
ビット4	バスドラム
ビット3	スネアドラム
ビット2	タムタム
ビット1	トップシンバル
ビット0	ハイハット

戻り値

なし

変更レジスタ

すべて

RMA_MK

コード

22

機　能

ミュージックキーボードをスキャンし、その状態を返します。

コール手順

DE　　バッファ（9バイト）のアドレス

IY　　マスター、スレーブどちらのキーボードを読むかをMIDBのアドレスで指定します。

戻り値

指定されたバッファに以下のように状態が返されます。

表7.72　ミュージックキーボードバッファの内容一覧

		ビット							
		7	6	5	4	3	2	1	0
オフセット	0	0	C	B	A#	0	A	G#	G
	1	0	F#	F	E	0	D#	D	C#
	2	0	C	B	A#	0	A	G#	G
	3	0	F#	F	E	0	D#	D	C#
	4	0	C	B	A#	0	A	G#	G
	5	0	F#	F	E	0	D#	D	C#
	6	0	C	B	A#	0	A	G#	G
	7	0	F#	F	E	0	D#	D	C#
	8	0	C	0	0	0	0	0	0

キーが押されていればそのビットに1が入ります。オフセット8のCがオクターブ2、オフセット0のCがオクターブ6です。

変更レジスタ

すべて

RMA_LFO

コード

23

機　能

振幅変調やビブラートの深さを設定します。

コール手順

C　深さ指定

ビット7	振幅変調の深さの指定	
	0	1dB
	1	4.8dB
ビット6	ビブラートの深さの指定	
	0	7セント
	1	14セント
ビット5〜0	無効	

IY　マスター、スレーブどちらの Y8950 に対して設定するかを MIDB のアドレスで指定します。

戻り値

なし

変更レジスタ

すべて

RMA_TRANS

コード

24

機　能

現在発声中の音および以降の音のトランスポーズを設定します。

コール手順

DE　トランスポーズ値
単位は 100 / 256 セントです。

IY　マスター、スレーブどちらの Y8950 に対して設定するかを MIDB のアドレスで指定します。

戻り値

なし

変更レジスタ

すべて

注　意

この機能を呼び出した後に RM_PITCH (ピッチの設定) や RM_TSRAN (トランスポーズの設定 2) を呼び出すと後からの設定のみが有効になります。

テンペラメント (音律)

MBIOS は平均律を扱っていますが、その平均律の音程をいろいろな音律の音程に変換することができます。1 オクターブ内の 12 音の各音程が、平均律の音程から何セント離れているかを 12

バイトのテーブルで記憶しています。テーブルのアドレスがワークエリアの3479Hにあり、ここは通常347BHを指しています。347BHからは実際のテーブルがあり、以下の音程と対応しています。

表7.73 テンペラメントテーブルの対応一覧

オフセット	コード
0	C
1	C#
2	D
3	D#
4	E
5	F
6	F#
7	G
8	G#
9	A
10	A#
11	B

テーブルの各1バイトは8ビットの符号付き（負数は2の補数表現）で、平均律の音程からのずれを0〜7FH（0から49.6セント）、80HからFFH（−50セントから−0.4セント）で指定します。

RM_UTEMPR

コード

28

機能

平均律の音程を設定されている音律の音程に変換します。

コール手順

D　　音程（中央Cが60）

戻り値

DE　　変換された音程

変更レジスタ

すべて

RM_CTEMPR

コード

29

機　能

音律を設定します。

コール手順

C　音律コード

0	ピタゴラス
1	ミーントーン
2	ヴェルクマイスター
3	ヴェルクマイスター（修正）
4	ヴェルクマイスター（別）
5	キルンベルガー
6	キルンベルガー（修正）
7	ヴァロッティ・ヤング
8	ラモー
9	完全平均律（初期値）
10	純正律　c メジャー（a マイナー）
11	純正律　cis メジャー（b マイナー）
12	純正律　d メジャー（h マイナー）
13	純正律　es メジャー（c マイナー）
14	純正律　e メジャー（cis マイナー）
15	純正律　f メジャー（d マイナー）
16	純正律　fis メジャー（es マイナー）
17	純正律　g メジャー（e マイナー）
18	純正律　gis メジャー（f マイナー）
19	純正律　a メジャー（fis マイナー）
20	純正律　b メジャー（g マイナー）
21	純正律　h メジャー（gis マイナー）

戻り値

なし

変更レジスタ

すべて

RM_PITCH

コード

30

解　説

現在発声中の音および以降の音のピッチを設定します。

コール手順

BC　マスターチャンネルのピッチ

DE　スレーブチャンネルのピッチ

ピッチは中央Cのすぐ上のAの周波数で指定します。範囲は410〜459で、単位はHzです。初期値は440Hzです。

戻り値

なし

変更レジスタ

すべて

注　意

この機能は内部でRMA_TRANS(トランスポーズの設定1)を実行していますので、この機能とトランスポーズと同時に使用する場合には、次のRM_TSRANでトランスポーズを設定して下さい。

RM_TSRAN

コード

31

機　能

現在発声中の音および以降の音のトランスポーズを設定します。

コール手順

BC　マスターチャンネルのトランスポーズ値

DE　スレーブチャンネルのトランスポーズ値

トランスポーズ値は-12799〜12799の範囲で指定します。単位はセントで、初期値は0です。

戻り値

なし

変更レジスタ

すべて

■インスツルメント

インスツルメントに割り当てられたFM音源チャンネルのみに対する操作です。

RI_DAMP

コード

48

機　能

割り当てられたFM音源チャンネルすべてを強制的に停止します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

RI_ALLOFF

コード

49

機　能

割り当てられたFM音源チャンネルすべてをキーオフします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

RI_EVENT

コード

50

機　能

指定された音程を音律の変換をしてキーオン、またはキーオフします。

コール手順

■キーオンのとき

D　　　音程（中央Cが60）+80H

E　　　ベロシティ（0が最弱、15が最強）

■キーオフのとき

D　　　音程（中央Cが60）

戻り値

なし

変更レジスタ

すべて

注　意

インスツルメントは楽器をシミュレートしているので、同じ音程を2度続けてキーオンすると、その音が1度キーオフされて再びキーオンされます。

RI_PCHB

コード

51

機　能

ピッチベンダーの位置を与えます。

コール手順

DE　　ピッチベンダーの位置
　　　16ビットの符号付き（負数は2の補数表現）です。7FFFHが最上位置、0が中央、8000Hが最下位置です。

戻り値

なし

変更レジスタ

すべて

注　意

この機能は内部でRCA_PARAM（リアルタイムパラメータの設定）を呼び出しています。

RI_PCHBR

コード

52

機　能

ピッチベンダーが音程に与える度合を設定します。

コール手順

C　　度合（0〜12）
　　　単位は100セントです。例えばこの値を4にし、ピッチベンダーの位置を7FFFHにした場合、音程は約+400セント変化します。

戻り値

なし

変更レジスタ

すべて

RIA_PARAM

コード

53

機　能

割り当てられた FM 音源チャンネルにリアルタイムパラメータを設定します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。

C　リアルタイムパラメータの CHDB 先頭からのオフセット

DE　設定データ

この機能で設定できるパラメータは以下のとおりです。

YCA_TRANS
YCA_VOL

戻り値

なし

変更レジスタ

すべて

RIA_VOICE

コード

54

機　能

割り当てられた FM 音源チャンネルに音色番号でボイスを設定します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。

C　音色番号（0〜63）

戻り値

なし

変更レジスタ

すべて

RIA_VPARAM

コード

55

機　能

割り当てられたFM音源チャンネルにボイスパラメータを設定します。

コール手順

IY　　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。

C　　リアルタイムパラメータのCHDB先頭からのオフセット

DE　　設定データ

この機能で設定できるパラメータは以下のとおりです。

YCAO0_MULTI、YCAO1_MULTI
YCAO0_LS、YCAO1_LS
YCAO0_AR、YCAO1_AR
YCAO0_RR、YCAO1_RR
YCAO0_VELS、YCAO1_VELS
YCAO0_VTL、YCAO1_VTL
YCA_VTRANS
YCA_FB

戻り値

なし

変更レジスタ

すべて

RIA_VOICEP

コード

56

機 能

FM音源チャンネルに音色データでボイスを設定します。

コール手順

IY　　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。
BC　　音色データのアドレス

戻り値

なし

変更レジスタ

すべて

■ ADPCM・PCM

ADPCM・PCMの録音、再生、転送およびADPCM、PCM間での変換などを行います。これらの操作のためにアプリケーションはPDB（PCM Data Block）を用意し、IXレジスタによってその領域をポイントします。

ローカルメモリに対してADPCMで録音や再生を行う場合をローカルモードと呼び、Y8950がCPUの途中介在なしに録音・再生ができます。MBIOSはこのモードをサポートし、しかも、CPUに対して非同期割り込み機能を用いて録音や再生の終了を知らせることもできます。

ADPCMの録音・再生時には、操作対象となるデバイスによってエントリが以下のように異なることに注意して下さい。

■ローカルメモリに対して操作を行う場合

録音時　RMA_ADREC
再生時　RMA_ADPLAY

■内部メモリに対して操作を行う場合

録音時　RMA_ADREC_DI
再生時　RMA_ADPLAY_DI

ローカルモードでない録音・再生動作の最中に CTRL ＋ STOP キーが押された場合、その動作は中断されます。ただしMSXのワークエリア【BASROM (FBB1H)】が0でなければ中断されません。

■ ADPCM・PCM録音時の同期スタート

音声を録音するとき、通常はMBIOSが呼び出された時点から録音が開始されます。しかし、

同期スタートモードを指定すると、入力音声がある程度の大きさになってから録音を開始させることができます。同期スタートモードを指定するには、アプリケーションがMBIOSを呼び出す前に【SYNCRF(3488H)】に0でない値を書き込みます。

ADPCM・PCM再生時のリピート再生

音声を再生するとき、1つのデータを繰り返して再生することができます。この動作をリピート再生といいます。アプリケーションがMBIOSを呼び出す前に【REPFLG(3489H)】に0でない値を書き込むと、リピート再生ができます。

また、リピート再生を停止するには、モードによって以下の方法があります。

■ローカルモードの場合

RMA_BREAK（ローカルモード再生・録音の中断）を呼び出す。

■非ローカルモードの場合

CTRL + STOP キーを押す。

PDBの詳細

PDBは16バイトで構成されています。このデータブロックはアプリケーションプログラムがADPCM・PCM操作をするときにその領域を用意します。

表7.74 PDB内容一覧

オフセット	ラベル	意味
0	PDB_DEV	ADPCM・PCM操作の対象となるデータが保存されているデバイスを設定します。 0　マスターチャンネルローカルRAM 1　マスターチャンネルローカルROM 2　スレーブチャンネルローカルRAM 3　スレーブチャンネルローカルROM 4　メインRAM 5　VRAM
1	未使用	
2	PDB_ADDR(下位8ビット)	ADPCM・PCM操作の対象となるデータの開始アドレスを設定します。単位は256バイトですが、デバイスがメインRAMの場合に限り、1バイトになります。
3	PDB_ADDR(上位8ビット)	
4	PDB_SIZE(下位8ビット)	ADPCM・PCM操作の対象となるデータの長さを設定します。単位はデバイスに関わらず256バイトです。
5	PDB_SIZE(上位8ビット)	
6	PDB_SAMPLE(下位8ビット)	ADPCM・PCMの録音・再生時のサンプリング周波数を設定します。単位は1Hzです。範囲はADPCMのとき1800〜16000、PCMのとき1800〜12000です。
7	PDB_SAMPLE(上位8ビット)	
8	PDB_PCM(下位8ビット)	ADPCMのトレース時およびPCMへの変換をするときにADPCMの初期予測値として用いられます。この値はオフセットバイナリで表され、8000Hが0です。通常、8000H(つまり0)を設定します。
9	PDB_PCM(上位8ビット)	
10	PDB_STEP(下位8ビット)	ADPCMのトレース時およびPCMへの変換をするときにADPCMの初期量子化幅として用いられます。この値は絶対値で表され、範囲は007EH〜6000Hです。通常007FHを設定します。
11	PDB_STEP(上位8ビット)	
12〜15	未使用	

次に ADPCM・PCM の各機能について説明します。

RM_MOVE_DI

コード

0

機　能

各デバイス間の ADPCM・PCM データを転送します。

コール手順

IX　　転送元を示す PDB のアドレス
　　　転送元を示す PDB に以下の項目を設定します。
　　　　PDB_DEV（デバイス番号）
　　　　PDB_ADDR（開始アドレス）
　　　　PDB_SIZE（転送するサイズ）

IY　　転送先を示す PDB のアドレス
　　　転送先を示す PDB に以下の項目を設定します。
　　　　PDB_DEV（デバイス番号）
　　　　PDB_ADDR（開始アドレス）

戻り値

CY フラグ　転送元、転送先のどちらであれ、指定されたデバイスがローカルモードによる動作中である場合には、転送は行われず 1 にセットされ戻ります。転送が行われた場合、転送元の PDB_SIZE が転送先の PDB_SIZE にコピーされます。

変更レジスタ

すべて

RM_READ_DI

コード

26

機　能

デバイスの ADPCM・PCM データをメイン RAM に 256 バイト転送します。

コール手順

IX　　転送元を示す PDB のアドレス
DE　　転送先のメモリアドレス
　　　転送元を示す PDB に以下の項目を設定します。
　　　　PDB_DEV（デバイス番号）
　　　　PDB_ADDR（開始アドレス）

戻り値

CY フラグ　転送先のデバイスがローカルモードによる動作中である場合には、行われず1がセットされ戻ります。

変更レジスタ

すべて

RM_WRITE_DI

コード

27

機　能

メイン RAM の 256 バイトの ADPCM・PCM データをデバイスに転送します。

コール手順

DE　　転送元のメモリアドレス
IY　　転送先を示す PDB のアドレス
　　　転送先を示す PDB に以下の項目を設定します。
　　　　PDB_DEV（デバイス番号）
　　　　PDB_ADDR（開始アドレス）

戻り値

CY フラグ　転送元のデバイスがローカルモードによる動作中である場合には、転送は行われず1がセットされ戻ります。

変更レジスタ

すべて

RM_TRACE_DI

コード

1

機　能

初期の予測値と量子化幅を元に ADPCM データをトレースし、次の予測値と量子化幅を求めます。

コール手順

C　　開始モード

0　　初期予測値を 8000H、量子化幅を 007FH で開始します。

1　　初期予測値、量子化幅は PDB の値を使用します。

IX　　トレースするデータを示す PDB のアドレス

PDB に以下の項目を設定します。

PDB_DEV（デバイス番号）

PDB_ADDR（開始アドレス）

PDB_SIZE（トレースサイズ）

開始モードが 1 の場合、次の項目も設定します。

PDB_PCM（初期予測値）

PDB_STEP（初期量子化幅）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中である場合には、トレースは行われず 1 がセットされ戻ります。

トレースが行われた場合、PDB に以下の項目のデータが設定されます。

PDB_ADDR（次の開始アドレス）

PDB_PCM（次の予測値）

PDB_STEP（次の量子化幅）

これらのデータを次回のトレースに使用して、大きな ADPCM のデータを幾度かに分割してトレースすることができます。

変更レジスタ

すべて

RM_CONV_PCM_DI

コード

2

機 能

初期の予測値と量子化幅を元に ADPCM のデータを PCM のデータに変換します。

コール手順

C　開始モード

　0　初期予測値を 8000H、量子化幅を 007FH で開始します。

　1　初期予測値、量子化幅は PDB の値を使用します。

IX　変換元の ADPCM データを示す PDB のアドレス

IY　変換先の PCM データを示す PDB のアドレス

変換元の PDB に以下の項目を設定します。

　PDB_DEV（デバイス番号）

　PDB_ADDR（開始アドレス）

　PDB_SIZE（変換量）

　PDB_SAMPLE（サンプリング周波数）

開始モードが 1 の場合、次の項目も設定します。

　PDB_PCM（初期予測値）

　PDB_STEP（初期量子化幅）

変換先の PDB に以下の項目を設定します。

　PDB_DEV（デバイス番号）

　PDB_ADDR（開始アドレス）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中である場合には、トレースは行われず、1 がセットされて戻ります。変換が行われた場合、PDB に以下の項目のデータが返されます。

■変換元の PDB

　PDB_PCM（次の予測値）

　PDB_STEP（次の量子化幅）

これらのデータを次回の変換に使用して、大きな ADPCM のデータを幾度かに分割して変換することができます。

■変換先の PDB

PDB_SIZE（変換後の量）

PDB_SAMPLE（変換元のサンプリング周波数のコピー）

変更レジスタ

すべて

注　意

この変換はデータ量が2倍になるので、変換先のデータ領域の大きさに注意して下さい。

RM_CONV_ADPCM_DI

コード

3

機　能

初期の予測値と量子化幅を元に PCM のデータを ADPCM のデータに変換します。

コール手順

C　開始モード

0　初期予測値を 8000H、量子化幅を 007FH で開始します。

1　初期予測値、量子化幅は PDB の値を使用します。

IX　変換元の PCM データを示す PDB のアドレス

IY　変換先の ADPCM データを示す PDB のアドレス

変換元の PDB に以下の項目を設定します。

PDB_DEV（デバイス番号）

PDB_ADDR（開始アドレス）

PDB_SIZE（変換量）

PDB_SAMPLE（サンプリング周波数）

変換先の PDB に以下の項目を設定します。

PDB_DEV（デバイス番号）

PDB_ADDR（開始アドレス）

開始モードが1の場合、次の項目も設定します。

PDB_PCM（初期予測値）

PDB_STEP（初期量子化幅）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中である場合には、トレースは行われず 1 がセットされて戻ります。変換が行われた場合、変換先の PDB に以下の項目のデータが返されます。

PDB_SIZE（変換後の量）
PDB_SAMPLE（変換元のサンプリング周波数のコピー）
PDB_PCM（次の予測値）
PDB_STEP（次の量子化幅）

これらのデータを次回の変換に使用して、大きな ADPCM のデータを幾度かに分割して変換することができます。

変更レジスタ

すべて

注　意

この変換はデータ量が 2 分の 1 になるので、変換元のデータ領域の大きさは偶数でなればなりません。

RMA_DAC_BIAS

コード

4

機　能

PCM 再生を行うときの音量（Y8950 のレジスタ 17H）を設定します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　音量（指数）
範囲は 1〜7 で、7 が最大音量です。

戻り値

なし

変更レジスタ

すべて

RMA_DAC_DI

コード

5

解　説

PCM データを再生します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）
IX　再生・録音の対象となる PDB のアドレス
PDB の以下の項目を設定します。
PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）
PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中である場合には、動作は行われず 1 がセットされて戻ります。

変更レジスタ

すべて

RMA_ADC_DI

コード

6

解　説

PCM データを録音します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）
IX　再生・録音の対象となる PDB のアドレス
PDB の以下の項目を設定します。

PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）
PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中である場合には、動作は行われず1がセットされて戻ります。

変更レジスタ

すべて

RMA_ADPCM_BIAS

コード

7

機　能

ADPCM 再生を行うときの音量を設定します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　音量
範囲は 0〜63 で、63 が最大音量です。

戻り値

なし

変更レジスタ

すべて

RMA_ADPLAY_DI

コード

8

機 能

ADPCM データを非ローカルモードで再生します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）
IX　再生・録音の対象となる PDB のアドレス
PDB の以下の項目を設定します。

PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）
PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中であったり、外部デバイスである場合には、動作は行われず 1 がセットされて戻ります。

変更レジスタ

すべて

RMA_ADREC_DI

コード

9

機 能

ADPCM データを非ローカルモードで録音します。

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）
IX　再生・録音の対象となる PDB のアドレス
PDB の以下の項目を設定します。

PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）
PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中であったり、外部デバイスである場合には、動作は行われず1がセットされて戻ります。

変更レジスタ

すべて

RMA_ADPLY_SAMPLE

コード

43

解　説

ローカルモードの再生を行っている最中にサンプリング周波数を変更します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。
DE　サンプリング周波数

戻り値

なし

変更レジスタ

すべて

RMA_BREAK

コード

10

機　能

ローカルモードの再生や録音の動作を中断します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。

戻り値

なし

変更レジスタ

すべて

RMA_ADPLAY

コード

11

解　説

ADPCM データをローカルモードで再生します。

コール手順

IY　　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。

C　　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）

IX　　再生・録音の対象となる PDB のアドレス

PDB の以下の項目を設定します。

PDB_DEV（デバイス番号）

PDB_ADDR（開始アドレス）

PDB_SIZE（サイズ）

PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　　指定されたデバイスがローカルモードによる動作中であったり、非外部デバイスである場合には、動作は行われず 1 がセットされて戻ります。

変更レジスタ

すべて

RMA_ADREC

コード

12

解　説

ADPCMデータをローカルモードで録音します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。
C　フィルタ指定（表7.79中のZMA_PH_FILTER参照）
IX　再生・録音の対象となるPDBのアドレス
PDBの以下の項目を設定します。
PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）
PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CYフラグ　指定されたデバイスがローカルモードによる動作中であったり、非外部デバイスである場合には、動作は行われず1がセットされて戻ります。

変更レジスタ

すべて

RMA_ADPLAYLP

コード

42

機　能

ADPCMデータをローカルモードで再生します。データの終わりまで再生が終了すると先頭から再び再生を開始し、中断されるまでその動作を続けます。中断するにはRMA_BREAK（ローカルモード再生・録音の中断）を使用します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。
C　フィルタ指定（表7.79中のZMA_PH_FILTER参照）
IX　再生の対象となるPDBのアドレス
PDBの以下の項目を設定します。
PDB_DEV（デバイス番号）
PDB_ADDR（開始アドレス）

PDB_SIZE（サイズ）
PDB_SAMPLE（サンプリング周波数）

戻り値

CY フラグ　指定されたデバイスがローカルモードによる動作中であったり、非外部デバイスである場合には、動作は行われず1がセットされて戻ります。

変更レジスタ

すべて

RMA_PHASE_SET_DI

コード

13

機　能

メイン RAM 内の 256 バイトの PCM データを 1 つの波形データと見なし、以下のような ADPCM データに変換して外部 RAM 内に記憶します。

外部 RAM アドレス	音程番号	波形数
0000H～07FFH	24H～36H	16個
0800H～0FFFH	37H～42H	32個
1000H～17FFH	43H～4EH	64個
1800H～1FFFH	4FH～5AH	128個

コール手順

IY　マスター・スレーブどちらのチャンネルかを MIDB のアドレスで指定します。
C　フィルタ指定（表 7.79 中の ZMA_PH_FILTER 参照）
DE　PCM データのアドレス

戻り値

なし

変更レジスタ

すべて

注　意

変換に先立って、RMA_BREAK（ローカルモード再生・録音の中断）が実行されます。

RMA_PHASE_EG

コード

14

機　能

エンベロープデータを設定します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。
DE　エンベロープデータ（7バイト）のアドレス
エンベロープデータの内容（「4.4.5 ワークエリアの詳細　サンプリングキーボードシミュレーション」参照）

オフセット	内　容
0	タイマ1に設定する値
1	トータルレベル
2	アタックレート
3	ディケイレート1
4	サスティンレベル
5	ディケイレート2
6	リリースレート

戻り値

なし

変更レジスタ

すべて

RMA_PHASE_EVENT

コード

15

機　能

指定された音程をサンプリングキーボードシミュレーションでキーオン、またはキーオフします。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。

■キーオンのとき

D　音程（中央Cが60）+80H

■キーオフのとき

D　音程（中央Cが60）

戻り値

なし

変更レジスタ

すべて

注　意

使用できる音程はキーコードで24H〜5AHです。

RMA_CSM_DI

コード

25

機　能

CSMデータを再生します。

コール手順

IY　マスター・スレーブどちらのチャンネルかをMIDBのアドレスで指定します。

B　再生音量
　範囲は0〜127で、0が最大音量です。

C　フィルタ指定（表7.79中のZMA_PH_FILTER参照）

DE　再生の対象となるCSMデータのアドレス

戻り値

なし

変更レジスタ

すべて

■ CSM データ

1つのデータはいくつかのブロックデータからなり､1つのブロックデータはいくつかのフレームデータからなります。

CSM データ

ブロック 1
ブロック 2
⋮
ブロック N (最終ブロック)

→ブロックデータ

00H 80H	最終ブロック以外 または 最終ブロック
	インターバル[*1]
	フレーム 1
	フレーム 2
	⋮
	フレーム N (最終フレーム)

→フレームデータ

00H 80H	最終フレーム以外 または 最終フレーム
	ピッチ[*2]
	トータルレベル 0[*3]
	トータルレベル 1
	トータルレベル 2
	トータルレベル 3
	NOTE 0[*4]
	NOTE 1
	NOTE 2
	NOTE 3

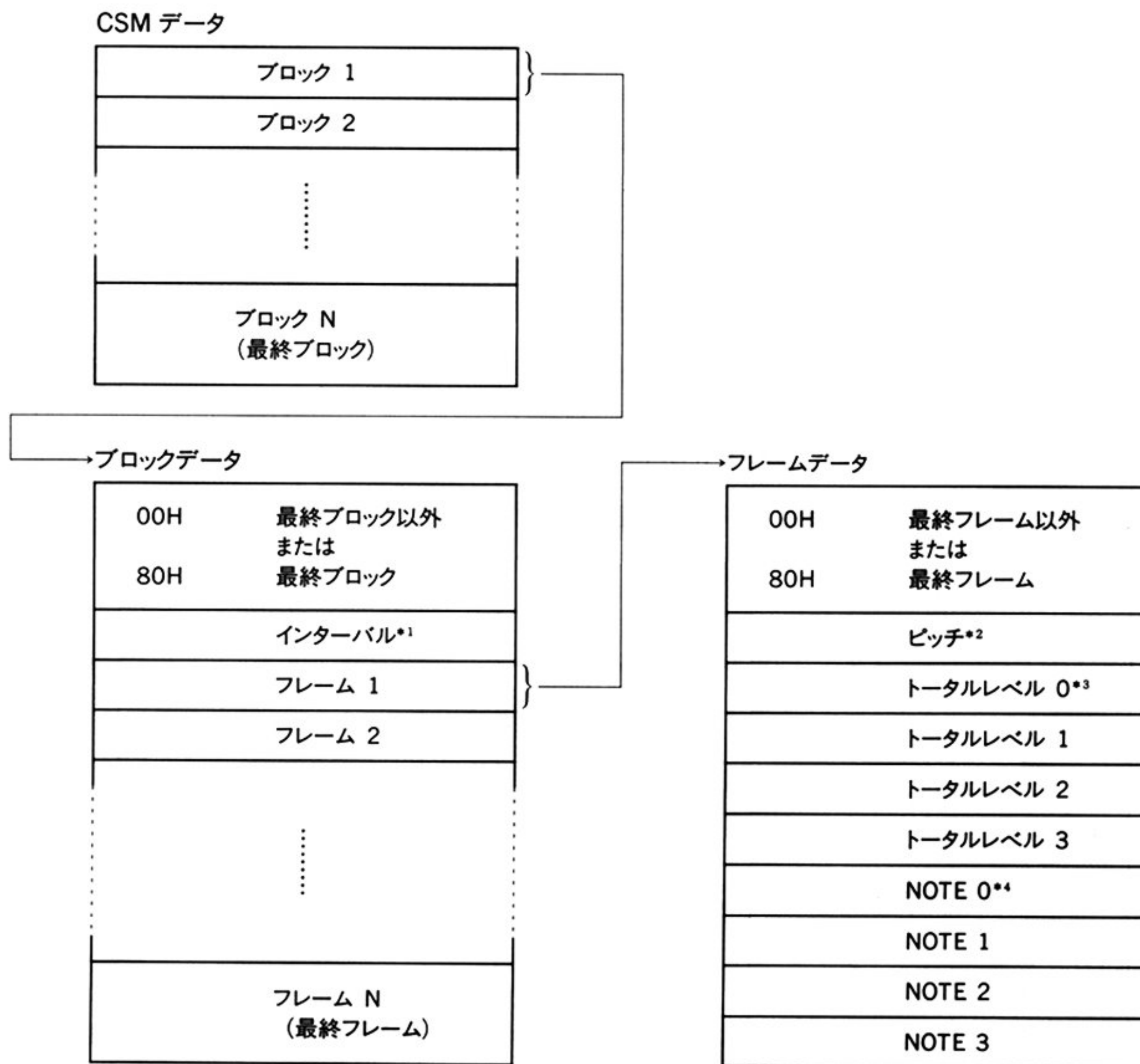



図 7.36 CSM データの形式

[*1] インターバル

タイマ2に設定されるデータで、0が81.9mS、255が0.32mSです。

[*2] ピッチ

タイマ1に設定されるデータで、0が20.9mS、255が0.08mSです。

[*3] トータルレベル

各チャンネルの音量データです。0が最大音量で127が最小音量です。

[*4] NOTE

各チャンネルの音程データです。上位4ビットでオクターブを0〜7の範囲で、下位4ビットで音階を指定します。

音階データ

0	C#
1	D
2	D#
3	なし
4	E
5	F
6	F#
7	なし
8	G
9	G#
10	A
11	なし
12	A#
13	B
14	C
15	なし

8. ミュージックキーボードのスキャン

SV_MK (00B1H)

機　能

ミュージックキーボードをスキャンして、前回のスキャン結果と比較し変化があればAST_MK（ミュージックキーボード非同期割り込み）を呼び出します。

コール手順

なし

戻り値

CY フラグ　非同期割り込み機能によって呼び出されたプログラムからの呼び出しの場合、スキャンはされずに1がセットされます。

変更レジスタ

すべて

注　意

マスターチャンネルとスレーブチャンネルのキーボードの状態は論理的にORがなされています。

9. 割り込みサービス

SV_IRQ (00B4H)

機　能

アプリケーションプログラムがハードウェアの割り込みを検出したときに（【H.KEYI (FD9AH)】を設定することにより可能)、MBIOSに対してそれを知らせるためのエントリです。このエントリの内部でY8950に対する割り込み処理を行い、非同期割り込みなどのサービスを実現しています。

コール手順

なし

戻り値

なし

変更レジスタ

なし

注　意

このエントリが呼ばれない限り、MBIOSはすべての割り込みサービスを行うことはできません。ハードウェアの割り込みによりアプリケーションに制御が渡った時点ではY8950のステータスレジスタはCPUに読まれていないので、割り込みのリクエストはかかったままなので、ハードウェア割り込みを許可する命令（EIなど）を実行してはなりません。

■ MBIOSの割り込み処理

MBIOSにおいて割り込みは次のように処理されます。

1. ハードウェアの割り込みによって0038H番地が呼ばれる。
2. 0038H番地からHK_IRQ38にジャンプする。

3. HK_IRQ38 は SV_IRQ をコールする。
4. SV_IRQ では、
   - MSX-AUDIO のステータスを読み込んで、リクエストのあった割り込みの割り込み要求回数を1増やす。
   - 次に HK_VDPIRQ をコールし、帰ってきたときのキャリーフラグが0であれば VDP のステータスを読み、割り込みリクエストがあれば割り込み要求回数を1増やす。
   - 次に HK_USERIRQ をコールする。帰ってきたらユーザー割り込みが許可されているか調べ、許可されていれば（ユーザー割り込みカウントが0のとき）UISV (User Interrupt SerVice) をコールする。禁止されていたら6.へ行く。
5. UISV では次の順に割り込み要求回数を調べ、1番最初にその回数が0でないベクタを見つけるとそのベクタで示されているプログラムをコールし（この際に割り込み要求回数が A レジスタで渡されます）、戻る。
   ADPCM 終了（マスターチャンネル）ベクタ
   ADPCM 終了（スレーブチャンネル）ベクタ
   IRQ_PRIO1 で示されるベクタ
   IRQ_PRIO2 で示されるベクタ
   IRQ_PRIO3 で示されるベクタ
   IRQ_PRIO4 で示されるベクタ
6. SV_IRQ の中に戻る。
7. HK_IRQ38 の中に戻る。
8. ハードウェア割り込みがかかったプログラムに戻る。

なお、MBIOS のスロットが表に出ていないときは MSX の BIOS にはいるため、アプリケーションは H.KEYI のフックを使用して MBIOS の 0038H 番地を呼び出さなければなりません。

■拡張 BASIC の割り込み使用法

MSX-AUDIO の拡張 BASIC において割り込み処理は次のように設定されます。

1. H.KEYI を設定してすべてのハードウェア割り込みを拡張 BASIC のプログラムに向けている。そのプログラムの中から MBIOS の SV_IRQ をコールする。
2. HK_IRQ38 を拡張 BASIC のプログラムに向けている。そのプログラムはレジスタをすべてセーブした後、MSX の BIOS の 0038H 番地をインタースロットコールする。
3. ADPCM 終了割り込みを拡張 BASIC に向けている。
4. IRQ_PRIO は初期値で使用。

5. HK_VRAM は初期値で使用。
6. HK_USERIRQ は初期値で使用。
7. HK_VDPIRQ はキャリーをセットするプログラムを設定。

■割り込み処理用ワークエリア

各々8バイトのフックエリアです。MBIOS の内部から呼び出されるプログラムが置かれており、以下のエントリがあります。

表 7.75　割り込み処理用ワークエリア一覧

ラベル	アドレス	機　能
HK_IRQ38	3300H	ハードウェア割り込みフック
HK_VDPIRQ	3308H	VDP 割り込み処理指定
HK_USERIRQ	3310H	アプリケーション割り込みフック
HK_VRAM	3318H	VRAM 操作フック
UISV ベクタテーブル	3340H〜	ユーザー割り込みサービスのベクタテーブル

HK_IRQ38（3300H）

機　能

MBIOS のスロットが表に出ていて割り込みがかかったときに、0038H 番地からこのフック（プログラム）にジャンプします。

SV_RESET で初期設定されるプログラム

```
hk_irq38:
        call    sv_irq
        ei
        ret
        defs    3
```

注　意

SV_RESET を呼び出した後にすぐ EI 命令を実行し、VDP からの割り込みがかかると暴走します。EI 命令を実行する前に、HK_IRQ38 を以下のように書き換えて下さい。

```
hk_irq38:
        call    sv_irq
        ret
        defs    4
```

HK_VDPIRQ (3308H)

機　能

SV_IRQ が呼ばれたときに VDP のステータスレジスタを読んでその処理をするかどうかを返すフック（プログラム）です。
このフックがリターンするときにキャリーフラグが 1 であるならば VDP のステータスは読まれません。0 であるならば読まれます。

SV_RESET で初期設定されるプログラム

```
hk_vdpirq:
        xor     a       ; clear carry flag
        ret
        defs    6
```

注　意

MSX の通常動作においては、VDP のステータスを MSX の BIOS 以外が読み込むと本体のキーボードなどの割り込み処理が行われなくなります。

HK_USERIRQ (3310H)

機　能

SV_IRQ が呼ばれたときにその中からアプリケーションプログラムを呼びたい場合に使用するフックです。

SV_RESET で初期設定されるプログラム

```
hk_userirq:
        ret
        defs    7
```

HK_VRAM (3318H)

機　能

ADPCM・PCM 操作において VRAM をアクセスする前に呼ばれるフックです。このフックがリターンするときにキャリーフラグが 1 であるならば VRAM に対する動作は行われません。0 であるならば行われます。

SV_RESET で初期設定されるプログラム

```
hk_vram:
        xor     a       ; clear carry flag
        ret
        defs    6
```

■ UISV ベクタテーブル

SV_IRQ が呼ばれて Y8950 のステータスを読んだ後、各々の割り込み原因別に呼び出すアプリケーションのアドレスのテーブルです。このアプリケーションを呼び出す機能を UISV（User Interrupt SerVice）と呼びます。MBIOS の内部を実行中は UISV は禁止され、ハードウェアの割り込みがあった場合にはその回数が記憶されます。UISV が許可され、次のハードウェア割り込みがあるとアプリケーションを呼ぶときに今までの回数を A レジスタで知らせます。各々のベクタは 3 バイトで構成され、最初の 1 バイトは割り込み回数の記憶に使われています。次の 2 バイトが呼び出すアプリケーションのアドレスです。

0000H が設定してあると呼び出しは行われません。

表 7.76　UISV ベクタテーブル一覧

ラベル	アドレス	内　容	初期値
IRQ_TIM1	3340H	タイマ 1 割り込み	0000H
IRQ_TIM2	3344H	タイマ 2 割り込み	SV_TEMPO
IRQ_VDP	3348H	VDP 割り込み	0000H
IRQ_USER	334CH	ユーザー割り込み	0000H
ZIRQ_ADP_M	3350H	ADPCM 終了割り込み（マスタ）	0000H
ZIRQ_ADP_S	3354H	ADPCM 終了割り込み（スレーブ）	0000H

■UISV 優先順位テーブル

1回のUISVで処理される割り込みは1つだけなので、同時に2つ以上の割り込みが起きたときにどの割り込みを優先させるかをこのテーブルに登録します。具体的には、UISVベクタテーブルのアドレスを保存しておきます。

表7.77 UISV優先順位テーブル一覧

ラベル	アドレス	優先順位	初期値
IRQ_PRIO1	3360H	1（最高優先）	IRQ_TIM2
IRQ_PRIO2	3362H	2	IRQ_TIM1
IRQ_PRIO3	3364H	3	IRQ_VDP
IRQ_PRIO4	3366H	4（最低優先）	IRQ_USER

■非同期割り込み

非同期割り込み機能はMBIOSにあらかじめ登録し、ある事象が起きたときにMBIOSからアプリケーションを呼び出す機能です。

MBIOSがアプリケーションを呼び出すときには、インタースロットコールではなく通常のコール命令が使われるので、MBIOSと同じCPUアドレス領域(0000H〜3FFFH)にある別のスロットのプログラムは呼び出せません。

アプリケーションが呼び出された時点では、MBIOSを直接コール命令で呼び出すことができます。アプリケーションからMBIOSに戻るときは通常のリターン命令を使用します。

非同期割り込みサービスには以下のようなものがあります。

表7.78 非同期割り込み一覧

ラベル	アドレス	機 能
AST_MK	3370H	ミュージックキーボード割り込み
AST_ADPCM_M	3372H	ADPCM終了割り込み
AST_ADPCM_S	3374H	ADPCM終了割り込み
SV_TEMPO	00B7H	自動キーオフ処理

ここでは以下のように説明します。

アドレス

MBIOSが呼び出すアドレスを記憶しているアドレスです。この内容が0のときは呼び出しは行われません。

コール手順

アプリケーションが呼び出されるときに設定される値です。

戻り値

アプリケーションがMBIOSに返す値です。

変更レジスタ

アプリケーションが破壊してもかまわないレジスタです。

AST_MK（3370H）

機　能

ミュージックキーボードの変化を検出したときに呼び出されます。アプリケーションがキーボードの記録などをするときに使用できます。

コール手順

■キーが押されたとき

D	7	6	5	4	3	2	1	0
	1	×	×	×	×	×	×	×

（ビット6〜0：キーコード）

E　キーを押した速さ

ただし、現在のMBIOSはキー速度検出機能を持っていないのでSM_MK（ミュージックキーボードスキャナーの初期化）で設定された値が入ります。

■キーが離されたとき

D	7	6	5	4	3	2	1	0
	0	×	×	×	×	×	×	×

（ビット6〜0：キーコード）

戻り値

なし

変更レジスタ

IX、IY以外のすべて

注　意

この機能によって呼び出されたアプリケーションは、以下のMBIOSエントリを呼び出すことはできません。

SV_MK（ミュージックキーボードのスキャン）
SV_SETUP（諸機能の初期設定）

AST_ADPCM_M（3372H）

機　能

マスターチャンネルで、外部デバイスを使用したADPCMの再生や録音が終了したときに呼び出されます。

コール手順

IX　　再生・録音に使用されたPDBのアドレス

戻り値

なし

変更レジスタ

IX以外のすべて

注　意

この機能によって呼び出されたアプリケーションは、以下のMBIOSエントリを呼び出すことはできません。

SV_MK（ミュージックキーボードのスキャン）
SV_SETUP（諸機能の初期設定）

AST_ADPCM_S（3374H）

機　能

スレーブチャンネルで、外部デバイスを使用したADPCMの再生や録音が終了したときに呼び出されます。

コール手順

IX　　再生・録音に使用された PDB のアドレス

戻り値

なし

変更レジスタ

IX 以外のすべて

注　意

この機能によって呼び出されたアプリケーションは、以下の MBIOS エントリを呼び出すことはできません。

SV_MK（ミュージックキーボードのスキャン）
SV_SETUP（諸機能の初期設定）

10. 自動キーオフ処理

SV_TEMPO（00B7H）

機　能

各々の FM 音源チャンネルの ZC_COUNT（デュレーションカウント）を調べ、0 以外のときには 1 減らします。そして、値が 0 になったときにそのチャンネルに対してキーオフします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

11. スロットハンドリング

MSXのBIOSにあるスロット操作関連のプログラムはMBIOSにも用意されています。これらはアドレス、入出力条件ともに標準のMSXのBIOSと同等です。

RDSLT
WRSLT
CALSLT
ENASLT
CALLF

4.4.5 ワークエリアの詳細

1. MIDBの詳細

MIDB（Master Instrument Data Block）はMSX-AUDIO全体をコントロールするためにマスターチャンネル用とスレーブチャンネル用の2つがあります。1つのMIDBは64バイトで構成されています。マスターチャンネル用はMIDB_M、スレーブチャンネル用はMIDB_Sという名称です。

表7.79でMIDBの内容について説明します。なお、オフセットとは、各MIDBの先頭からのオフセットを意味します。

表7.79 MIDBの内容一覧（その1）

名　称	ラベル	オフセット	意　味
タイマ1	YM_TIM1	2	マスターチャンネル用のY8950に設定されたタイマ1の値を保存します。タイマ割り込みはマスターチャンネルにしか設定されないので、スレーブチャンネルのMIDBにはこの値はありません。
タイマ2	YM_TIM2	3	マスターチャンネル用のY8950に設定されたタイマ2の値を保存します。タイマ割り込みはマスターチャンネルにしか設定されないので、スレーブチャンネルのMIDBにはこの値はありません。
ADPCM再生ボリューム	YMA_BIAS	18	ADPCM再生時の音量を保存します。範囲は0～63です。
フィルタ設定	YMA_AUDIO	25	ADPCM・PCM録音再生時のフィルタ設定を保存します。
		ビット7～4	無効
		ビット3	フィルタの設定 0　PCM/CSM音用フィルタ 1　FM音用フィルタ
		ビット2～0	無効
トランスポーズ	YMA_TRANS	32(下位8ビット)	音程を計算するとき、トランスポーズ値として用いられます。全チャンネルに対して有効です。
		33(上位8ビット)	
LFO	YMA_LFO	34	Y8950に設定された振幅変調およびビブラートの深さを保存します。
		ビット7	振幅変調の深さ 0　1dB 1　4.8dB
		ビット6	ビブラートの深さ 0　7セント 1　14セント
		ビット5～0	無効

名 称	ラベル	オフセット	意 味
RAMサイズ	YMA_RAM	35	Y8950に接続されている外部RAMの容量をイニシャライズ時に256KbitのRAMがいくつあるかを調べて、その個数を保存します。
フラグ	ZMA_FLAG	36	そのチャンネルに関するフラグが保存されています。
		ビット7〜3	未使用
		ビット2	そのチャンネルに対するCHDB操作の無効フラグ 0　有効 1　無効
		ビット1	Y8950存在フラグ 0　このMIDB用のY8950は存在する 1　このMIDB用のY8950は存在しない
		ビット0	マスターチャンネル/スレーブチャンネル識別フラグ 0　マスターチャンネル用のMIDB 1　スレーブチャンネル用のMIDB
PDBアドレス	ZMA_PDB	37(下位8ビット)	ADPCM・PCM操作の際のPDB (PCM Data Block) のアドレスを記憶しています。
		38(上位8ビット)	
フィルタ指定	ZMA_PH_FILTER	39	ADPCM・PCM録音再生時のフィルタ指定を保存します。
		ビット7〜3	無効
		ビット2	自動フィルタ設定フラグ 0　発声対象によって自動的にフィルタを設定する 1　アプリケーションの設定を使用する
		ビット1	無効
		ビット0	フィルタの設定(ビット2が0のときは無効) 0　フィルタ特性A ADPCM・PCM録音再生のときに使用するフィルタ 1　フィルタ特性B FM音再生のときに使用するフィルタ

■サンプリングキーボードシミュレーション

サンプリングキーボードシミュレーションとは、PCMの波形データをADPCMデータに変換して、エンベロープをつけながら音を発声する仕組みです。エンベロープはタイマ0を使って割り込みをかけ、その割り込み毎にADPCM再生音量 (Y8950のレジスタ12H) を書き換えて実現しています。

表7.80はそれに関するデータの内容です。このデータはMIDBの後半部分です。

表7.80 MIDBの内容一覧（その2）

名 称	ラベル	オフセット	意 味
トータルレベル	ZMA_PH_TL	40	再生時のレベル（音量）を保存します。範囲は0～63で、63が最大レベルです。
アタックレート	ZMA_PH_AR	41(下位8ビット)	アタックレートを保持しています。タイマ割り込み時、そのときのレベルにこのアタックレートが加算され、和の上位8ビットがY8950に対して設定されます。その和の上位8ビットが63以上となったときはディケイレート1の処理に入ります。
		42(上位8ビット)	
ディケイレート1	ZMA_PH_D1R	43(下位8ビット)	ディケイレート1を保存します。タイマ割り込み時、そのときのレベルにこのディケイレート1が加算され、和の上位8ビットがY8950に対して設定されます。その和の上位8ビットが次のサスティーンレベルになったときはディケイレート2の処理に入ります。
		44(上位8ビット)	
サスティーンレベル	ZMA_PH_SL	45	ディケイレート1からディケイレート2に移るレベルであるサスティーンレベルを保持しています。範囲は0～63で、63が最大レベルです。
ディケイレート2	ZMA_PH_D2R	46(下位8ビット)	ディケイレート2を保存します。タイマ割り込み時、そのときのレベルにこのディケイレート2が加算され、和の上位8ビットがY8950に対して設定されます。その和の上位8ビットが0になるか、キーオフのイベントがあるまで割り込み毎に加算されます。
		47(上位8ビット)	
リリースレート	ZMA_PH_RR	48(下位8ビット)	リリースレートを保存します。タイマ割り込み時、そのときのレベルにこのリリースレートが加算され、和の上位8ビットがY8950に対して設定されます。その和の上位8ビットが0になるまで割り込み毎に加算されます。
		49(上位8ビット)	
カレントレベル	ZMA_PH_EG	50(下位8ビット)	エンベロープジェネレータを実現するため、そのときのレベルを保持するために使用されます。
		51(上位8ビット)	
エンベロープステータス	ZMA_PH_STAT	52	タイマ割り込み時に、以上に述べたどのレートを使用するかのステータスとして使用します。 0　音が発声されていない状態 1　アタックレート処理中 2　ディケイレート1処理中 3　ディケイレート2処理中 4　リリースレート処理中

2. CHDBの詳細

CHDB (CHannel Data Block) はFM音源のオペレータをコントロールするためのデータブロックで、チャンネルの数と同じく9つあり、Y8950に設定したデータを保存します。これは、Y8950の各レジスタが書き込み専用であるためです。

1つのCHDBは64バイトで構成され、その内32バイトはマスターチャンネル用で、残りの32バイトはスレーブチャンネル用です。各32バイトのうち、最初の8バイトはオペレータ0の、次の8バイトはオペレータ1のコントロールに使用されます。残りの16バイトは各オペレータに共通のデータに使用されます。

表7.81でCHDBの内容について説明します。なお、オフセットとは、各CHDBの先頭からのオフセットを意味します。

表7.81 CHDBの内容一覧

名 称	ラベル	オフセット	意 味	
マルチプル	YCAO0_MULTI	(operator 0)	0	
	YCAO1_MULTI	(operator 1)	8	Y8950のレジスタ番号20H(チャンネル0)〜35H(チャンネル8)に設定されている下記のようなデータを保存します。
		ビット7	AM (振幅変調指定)	
			0	振幅変調なし
			1	振幅変調あり
		ビット6	VIB (ビブラート指定)	
			0	ビブラートなし
			1	ビブラートあり
		ビット5	EG-TYP (エンベロープタイプ)	
			0	減衰音
			1	持続音
		ビット4	KSR (キースケールレート)	
			0	キースケールレートなし
			1	キースケールレートあり
		ビット3〜0	MULTIPLE (マルチプル) 実際に発生される周波数とB-number、F-numberとの倍率です。この値が大きくなれば、同じB-number、F-numberでも実際に発生される周波数は高くなります。	
			0	0.5倍
			1〜10	1〜10倍
			11	10倍
			12	12倍
			13	12倍
			14	15倍
			15	15倍
レベルキースケール	YCAO0_LS	(operator 0)	1	

名　称	ラベル	オフセット	意　味	
	YCAO1_LS	(operator 1)	9	Y8950のレジスタ番号40H (チャンネル0) ～55H (チャンネル8) に設定されているデータを保存します。ただし下位6ビットは無効で、上位2ビットのKSLだけが意味を持ちます。
		ビット7、6	KSL (レベルキースケール)	
		ビット5～0	無効	
アタック、ディケイレート	YCAO0_AR	(operator 0)	2	
	YCAO1_AR	(operator 1)	10	Y8950のレジスタ番号60H (チャンネル0) ～75H (チャンネル8) に設定されているデータを保存します。
		ビット7～4	AR (アタックレート)	
		ビット3～0	DR (ディケイレート)	
サスティンレベル、リリースレート	YCAO0_RR	(operator 0)	3	
	YCAO1_RR	(operator 1)	11	Y8950のレジスタ番号80H (チャンネル0) ～95H (チャンネル8) に設定されているデータを保存します。
		ビット7～4	SL (サスティンレベル)	
		ビット3～0	RR (リリースレート)	
ベロシティセンシティビティ	YCAO0_VELS	(operator 0)	4	
	YCAO1_VELS	(operator 1)	12	MBIOSではY8950が持っていない音の強弱をソフトウェアで実現していますが、その強弱の指定に対する実際の強弱の変化の度合を保存します。この値が大きければ、強弱の指定がより有効となります。小さければ、実際の強弱の変化が少なくなり、0の場合は強弱の指定は無効となります。
		ビット7～4	無効	
		ビット3～0	ベロシティセンシティビティ	
トータルレベル	YCAO0_VTL	(operator 0)	5	
	YCAO1_VTL	(operator 1)	13	Y8950のレジスタ番号40H (チャンネル0) ～55H (チャンネル8) に設定されるデータの基になるトータルレベルを保存します。ただし下位6ビットのみが意味を持ち、上位2ビットは無効です。音量の計算方法は下記を参照して下さい。
		ビット7、6	無効	
		ビット5～0	トータルレベル	
未使用		6	未使用領域です。	
未使用		7	未使用領域です。	
未使用		14	未使用領域です。	
未使用		15	未使用領域です。	

名　称	ラベル	オフセット	意　味
ボイストランスポーズ	YCA_VTRANS	16(下位8ビット) 17(上位8ビット)	音程を計算するとき、トランスポーズ値として用います。また、固定音程の音色データはこの値のみが Y8950 に設定されています。音程の計算方法は下記を参照して下さい。
トランスポーズ	YCA_TRANS	18(下位8ビット) 19(上位8ビット)	音程を計算するとき、トランスポーズ値として用いられます。音程の計算方法は下記を参照して下さい。
トリガ	YCA_TRIG	20	Y8950 のレジスタ番号 B0H（チャンネル 0）～B8H（チャンネル 8）に設定されているデータのうち鍵盤の ON・OFF のみをビット 5 に記憶しています。その他のビットは無効ですが 0 になっています。
ボリューム	YCA_VOL	21	Y8950 のレジスタ番号 40H（チャンネル 0）～55H（チャンネル 8）に設定されるデータの基になるボリュームを記憶しています。ただし下位 6 ビットのみが意味を持ち、上位 2 ビットは無効です。音量の計算方法は下記を参照して下さい。
		ビット 7、6	無効
		ビット 5～0	ボリューム
フィードバック	YCA_FB	22	Y8950 のレジスタ番号 C0H（チャンネル 0）～C8H（チャンネル 8）に設定されているデータを保存します。
		ビット 7～5	無効
		ビット 4	固定音程音色 このビットは Y8950 にはなく、MBIOS で実現している固定音程音色であることが指定されていることを意味しています。 0　　通常の音色 1　　固定音程音
		ビット 3～1	フィードバック 0～7 の値でフィードバック量を保存
		ビット 0	コネクション 2 つのオペレータの結合を保存 0　　直列周波数変調モード 1　　並列サイン波合成モード
ベロシティ	YCA_VEL	23	MBIOS では Y8950 が持っていない音の強弱をソフトウェアで実現していますが、その強弱の指定を保存します。
		ビット 7～4	無効
		ビット 3～0	ベロシティ
ピッチ	YCA_PITCH	24(下位8ビット) 25(上位8ビット)	Y8950 のレジスタ番号 A0H（チャンネル 0）～A8H（チャンネル 8）と B0H（チャンネル 0）～B8H（チャンネル 8）に設定されるデータの元になる音程を保存します。音程の計算方法は下記を参照して下さい。
ボイス番号	YCA_VOICE	26	Y8950 に音色データを設定したときの音色番号が保存されます。

名　称	ラベル	オフセット	意　味
フラグ	ZCA_FLAG	27	そのチャンネルに関するフラグを保存します。
		ビット7〜3	無効
		ビット2	そのチャンネルに対する操作の無効フラグ 0　操作を有効にする 1　操作を無効にする
		ビット1	無効
		ビット0	チャンネル指定 0　そのチャンネルはマスターチャンネル 1　そのチャンネルはスレーブチャンネル
チャンネル番号	ZC_CH	28	そのチャンネルのチャンネル番号が0〜8の値で記憶されています。このデータは初期化のときに1度だけ設定されます。
オペレータレジスタオフセット番号	ZC_OP	29	Y8950の各オペレータにデータを書き込むとき、そのオペレータのレジスタ番号を指定するために使用されます。このデータは初期化のときに1度だけ設定されます。以下のデータで初期化されます。
		チャンネル0	00H
		チャンネル1	01H
		チャンネル2	02H
		チャンネル3	08H
		チャンネル4	09H
		チャンネル5	0AH
		チャンネル6	10H
		チャンネル7	11H
		チャンネル8	12H
デュレーションカウント	ZC_COUNT	30(下位8ビット) 31(上位8ビット)	FM音源チャンネルに対して発音を指示したときにデュレーション（鍵盤で表すと、鍵盤を押している時間を意味する）を設定しますが、その時間経過を記憶しています。この値はSV_TEMPO（自動キーオフ処理)が呼ばれるたびに減ってゆき、0になるとそのチャンネルに対してキーオフが発生されます。

■音程の計算方法

MBIOSがY8950に対して設定するBlock-NumberとF-Numberの計算方法を示します。

YCA_FBのビット4が0のとき（固定音程音色でないとき）音程を決定するデータは以下の4つです。

YCA_PICTH	ビット15	常に0
	ビット14〜8	Key Code Numberといい中央Cが60で、1変化するごとに半音変化します。

	ビット7〜0	半音以下の細かい音程の指定です。1変化するごとに100/256セント変化します。
YCA_TRANS		2の補数表現のデータで、単位は100/256セントです。
YCA_VTRANS		〃
YMA_TRANS		〃　(MIDBのトランスポーズ参照)

YCA_FBのビット4が1のとき (固定音程音色のとき) 音程を決定するデータは以下の1つのみです。

YCA_VTRANS	ビット15	常に0
	ビット14〜8	Key Code Numberといい中央Cが60で、1変化するごとに半音変化します。
	ビット7〜0	半音以下の細かい音程の指定です。1変化するごとに100/256セント変化します。

これらのデータからF-NumberとB-Numberを求めるプログラム「FM.BAS」(固定音色でなく、マルチプルが1のとき) がフロッピーディスクに入っていますので、参照して下さい。

■音量の計算方法

MBIOSがY8950に対して設定するトータルレベル計算方法を示します。
オペレータ0に対して音量を決定するデータは以下の4つです。

YCA_VEL
YCAO0_VELS
YCAO0_VTL
YCA_VOL (YCA_FBのビット0が0のときは無視される)

オペレータ1に対して音量を決定するデータは以下の4つです。

YCA_VEL
YCAO1_VELS
YCAO1_VTL
YCA_VOL

これらのデータからトータルレベルを求めるプログラム「TOTAL.BAS」がフロッピーディスクに入っていますので、ご参照下さい。

4.4.6 用語について

ADPCM

ADPCMはAdaptive Differential Pulse Coded Modulationの略で、MSX-AUDIOではPCM化したデータを基に予測値と比較を行い、その差分を量子化幅によってコード化し、音質の低下を防ぎながら記憶領域を少なくする方式です。MSX-AUDIOは量子化幅を符号ビット+3ビットでコード化します。

CSM

複合正弦波合成の略。拡張BASICはこのモードに設定することのみをサポートしています。

MBIOS

MSX-AUDIOのシステムソフトウェアは大きくわけて拡張BASICとMBIOSに区別することができます。MBIOSはMusic Basic Input Output Systemの略で、MSX-AUDIO LSIの機能を直接扱うソフトウェアです。MSX-AUDIO LSIの基本的な機能をサポートしているため他のアプリケーションソフトウェアはMBIOSに用意されている必要な機能を呼びだすことで容易にMSX-AUDIO LSIの機能を使うことができます。拡張BASICもこのMBIOSの機能を利用して動作しています。

MK

Music Keyboardの略です。

MSX-AUDIO

MSX-AUDIOは株式会社アスキーが開発したMSXパーソナルコンピュータのための音源LSIで、9チャンネルの2オペレータFM音源と1チャンネルの4ビットADPCM音源を持っています。ADPCMのために外部メモリと呼ぶ専用メモリを最大256Kバイトまで持つことができるためCPUの負担なしにADPCMの録音再生を行うことができます。

MSX-AUDIOをサポートするシステムソフトウェアではこのLSIの持つ機能を容易に引き出すための機能を提供しており、MSX、MSX2、MSX2+で動作します。

MSX-AUDIOのチャンネル

MSX-AUDIOのシステムソフトウェアは、2つまでのMSX-AUDIO LSIを扱うことが

できるようになっています。

番号の小さいスロットに置かれているMSX-AUDIO LSIが第1チャンネルに、あとの方のスロットに置かれているMSX-AUDIO　LSIが第2チャンネルに割り当てられ、MBIOS、拡張BASICのいくつかの機能はこのチャンネル番号により2つのMSX-AUDIO LSIを区別して扱うことができます。

PCM / ADPCM

PCMはPulse Coded Modulationの略で、MSX-AUDIOは8ビット符号付き（負数は2の補数表現）でコード化されたデータを扱います。ADPCMは適応差分PCMの略で、MSX-AUDIOは4bitのADPCMを扱います。拡張BASICはこれらの形式の音声の記録再生を、番号で指定する音声ファイルとして扱うことができます。ADPCMとPCMのデータ形式の変換をする機能も用意されています。

Y8950

MSX-AUDIO LSIです。

インスツルメント

本来、楽器という意味ですが、ここではミュージックキーボード(MK)と結合されたFM音源を意味します。MSX-AUDIOのFM音源の9つのチャンネルのうちいくつかを同一の音色設定にしておき、ミュージックキーボードのキーのオン、オフによってそれらのチャンネルをオン、オフすることによってMSX-AUDIOを通常の鍵盤楽器と同じように扱えます。このようにミュージックキーボードとFM音源を関連づけた仕組みのことを指します。キーボードからはキーのオン、オフとキーコード番号(中央Cが60に対応)およびプログラムにより設定されたキーを押したときの強さ（ベロシティ）がインスツルメントに送られます。これらを総称してイベントと呼び、あるタイミングで発生したイベントを記録したり再生したりすることをMK記録と呼びます。

インスツルメント

MSX-AUDIOのFM音源の9つのチャンネルのうちいくつかをひとまとめにして楽器をシミュレートする仕組みを指します。この仕組みを利用すると、音程を指定するだけでMBIOSが割り当てられた内から使用していないチャンネルを自動的に選択して発声します。

キーオフ

鍵盤楽器で押していた鍵盤を離すことに例えた、音声出力を停止する動作を示します。

キーオフをしても音はすぐには止まらず、ある時間余韻が残ります。

キーオン

鍵盤楽器で鍵盤を押すことに例えた、音声出力を開始する動作を示します。

ダンプ

音声出力をすぐに停止する動作を示します。キーオフとはダンプした瞬間に音が止まる点が異なります。

ベロシティ

キーオンするときに鍵盤が押された速さのことで、音の強弱を表し、FM 音源の音量と音色に影響します。0〜15 の 16 段階が用意されていますが、MSX-AUDIO のキーボードはタッチ速度を検出する機構を持たないので、常に同じ値を返します。

マスターチャンネル、スレーブチャンネル

MSX-AUDIO カートリッジは 1 つの MSX に 2 つまで実装することができます。1 つだけのときはそのカートリッジのことをマスターチャンネルといい、スレーブチャンネルは存在しません。2 つのときはスロット番号の若いスロットに実装されているカートリッジをマスターチャンネルといい、別のカートリッジをスレーブチャンネルといいます。スレーブチャンネル側にある拡張 BASIC や MBIOS などのプログラムは MSX 立ち上がり時のみ動作し、それ以外はすべてマスターチャンネル側のプログラムがスレーブチャンネル側の Y8950 も制御します。

音色ライブラリ

MSX-AUDIO のシステムソフトウェアはシステム組み込みの FM 音源の音色パラメータのデータを 64 種類持っています。これを音色ライブラリとよびます。64 種類のライブラリの内 0〜31 番目までは ROM に置かれているため変更できませんが、32〜63 番目のものはプログラムにより変更することができ、特にユーザー音色ライブラリと呼ぶことがあります。

システム音色ライブラリ（SVL）

SVL は System Voice Library の略で ROM におさめた FM 音源の音色データを意味します。FM 音源音色番号で 0 番〜31 番までが SVL です。データは ROM にあるので変更はできません。

ユーザー音色ライブラリ（UVL）

UVLはUser Voice Libraryの略でRAMにあるFM音源の音色データを意味します。FM音源音色番号で32番から63番までがUVLです。

音声ファイル

PCM/ADPCMの音声データは番号をつけたファイルとして取り扱います。このファイルを音声ファイルとよびます。

外部メモリ

MSX-AUDIO LSIがローカルにもつメモリのことをさします。

ADPCMの録音と再生をこのメモリを対象として行うときにはローカルモードと言い、CPUの介入を必要としないためバックグラウンド処理で扱うことができます。

4.5 Y8950（MSX-AUDIO）

4.5.1 MSX-AUDIOの概要

1. 概要

MSX-AUDIOは、MSX2のオプショナル音源として開発された音源LSIです。

このLSIは音源としてFM音源システムを採用しているため、リアルで迫力ある音作りができます。さらに、従来のFM音源LSIで使われている複合正弦波音声合成方式に加えて、ADPCMによる音声分析・合成機能を備えているため、容易に音声データの処理が可能になりました。また、この音声分析・合成回路に組み込まれているAD/DAの各変換器を単独で使うこともでき、アナログデータをもこのLSIで処理可能です。その他、このLSIには鍵盤処理のための入出力ポートや汎用のI/Oポートも備えてありますので、MSX-AUDIOが1個あれば、音に関するすべての処理が可能になります。

2. 特徴

- FM音源を採用し、リアルなサウンドを作ることが可能。YM3526とソフトウェアコンパチブル
- モード選択により9音同時発生あるいはメロディ音6音＋リズム音5音の2つのモードを選択可能
- ビブラート発振器、振幅変調発振器内蔵
- 4ビットADPCM音声分析・合成回路内蔵
- 複合正弦波音声合成が可能
- AD/DAコンバータ制御回路内蔵
- 外部メモリとしてROM、RAMともに256Kバイト接続可能（ADPCMデータ格納あるいはCPUの補助記憶）
- 鍵盤スキャニングのための8ビット入出力ポート内蔵
- 4ビットI/Oポート内蔵
- 入出力TTLコンパチブル
- Si-gate CMOS LSI
- 5V単一電源

3. FM方式の概略

FM方式とは、Frequency Modulation すなわち周波数変調の意味で、変調によって生じる高調波を楽音の合成に利用したものです。この方式は比較的簡単な回路で、非調和音も含む高い高調波成分を持つ波形を発生させることができ、しかも変調指数と高調波のスペクトル分布の対応が非常に自然であるため、自然楽器の合成音から電子楽器まで、幅広い音作りが可能ということが確認されています。

FM方式は以下の式のように4つのパラメータで表現されます。

$$F = A\sin(\omega ct + I\sin\ \omega mt)$$

式7.8

ここでは、Aは出力振幅、Iは変調指数、また ωc、ωm はそれぞれキャリア、モジュレータの各周波数です。この7.8式は次のように表現することもできます。

$$F = A[Jo(I)\sin\omega ct + J1(I)\ \{\sin(\omega c+\omega m)t - \sin(\omega c-\omega m)t\}\ + J2(I)\ \{\sin(\omega c + 2\omega m)t + \sin(\omega c-2\omega m)t + ...]$$

式7.9

ここで、Jn(I)はn次の第1種Bessel関数です。式7.9からわかるように各倍音の振幅は、変調指数のBessel関数で表現されることになり、式7.8によるFM音源は特定の楽音や効果音の合成に非常に有効となることがわかります。ただし、これでは高調波が一様に分布しないためString系の音源には不向きとなります。そこで、考え出されたのが式7.10で表されるfeedback FMという方式です。

$$F = A\sin(\omega ct + \beta F)$$

式7.10

ここでβは帰還率です。このfeedback FMでは、高調波スペクトルが鋸歯状波となりString系の音作りも可能となります。

以上のような、FM方式を実現するためには、次の3つの機能ブロックが必要です。

1. ωtを発生させるphase generator (PG)
2. 振幅Aや変調指数Iを時間関数として得るためのenvelop generator (EG)
3. sinテーブル (sin)

以上の3つの構成要素を組み合わせて1つのユニットとして考えると、先のFM方式は図7.37のように表すことができます。したがって、このユニット(オペレータセル:OP)の考え方を用

いれば、FM方式の音作りは、ユニット内の周波数パラメータやEGパラメータの設定、そしてユニット間の組み合せのデータを作ればよいことになります。

a. $F(t)=A\ SIN\ (\omega ct+I\ SIN\ \omega mt)$

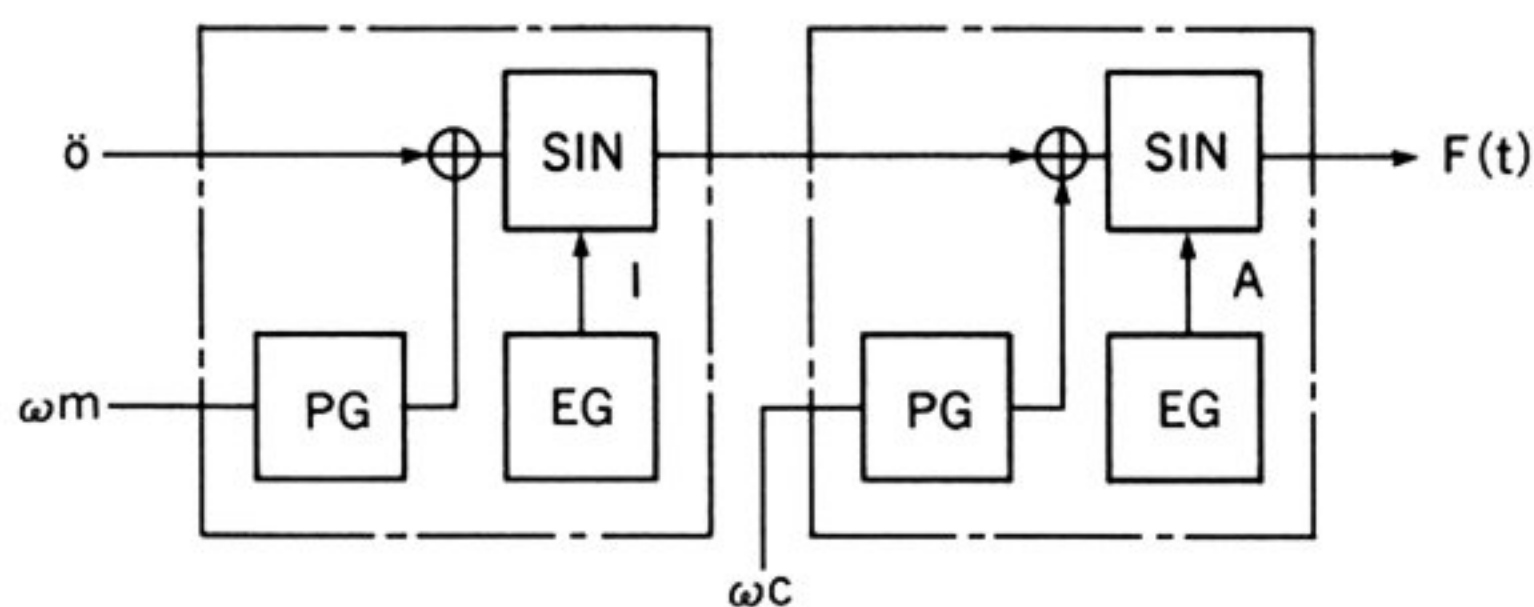
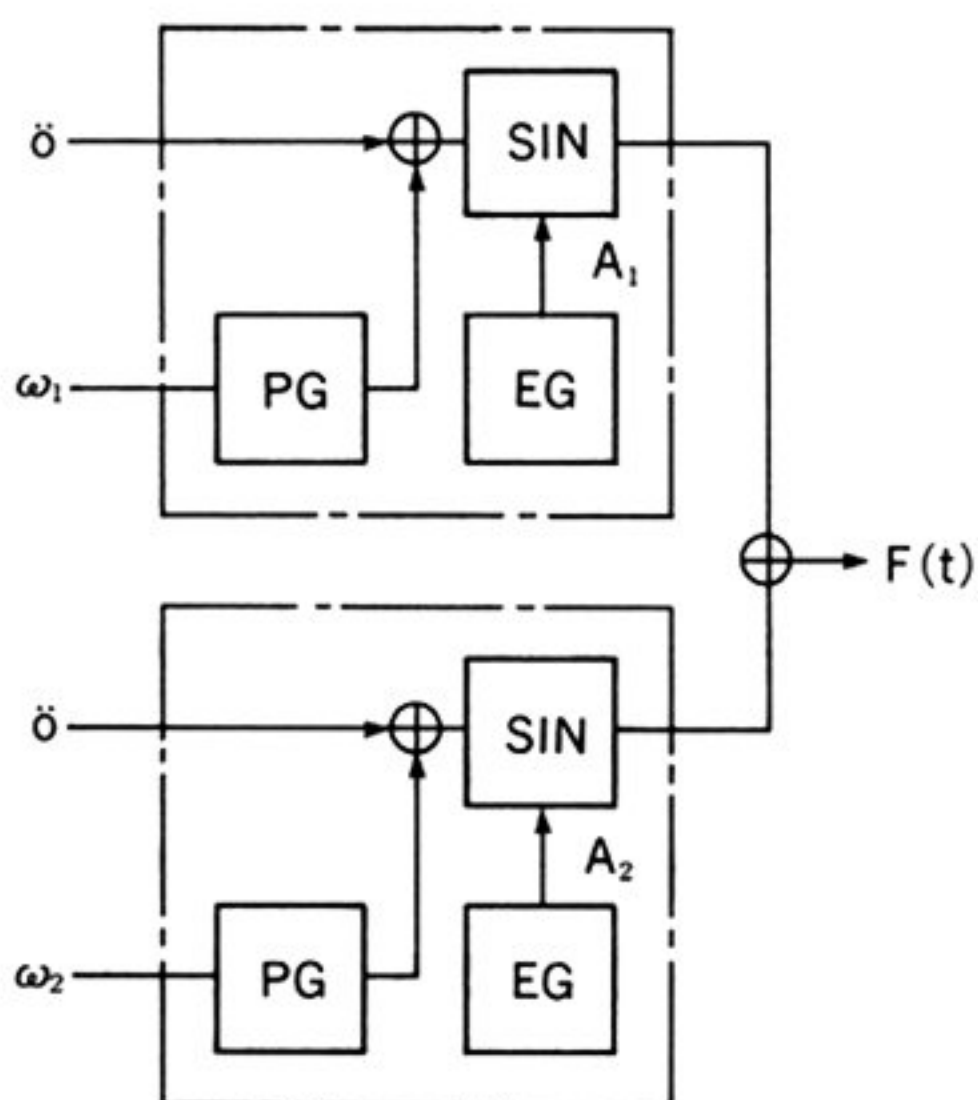
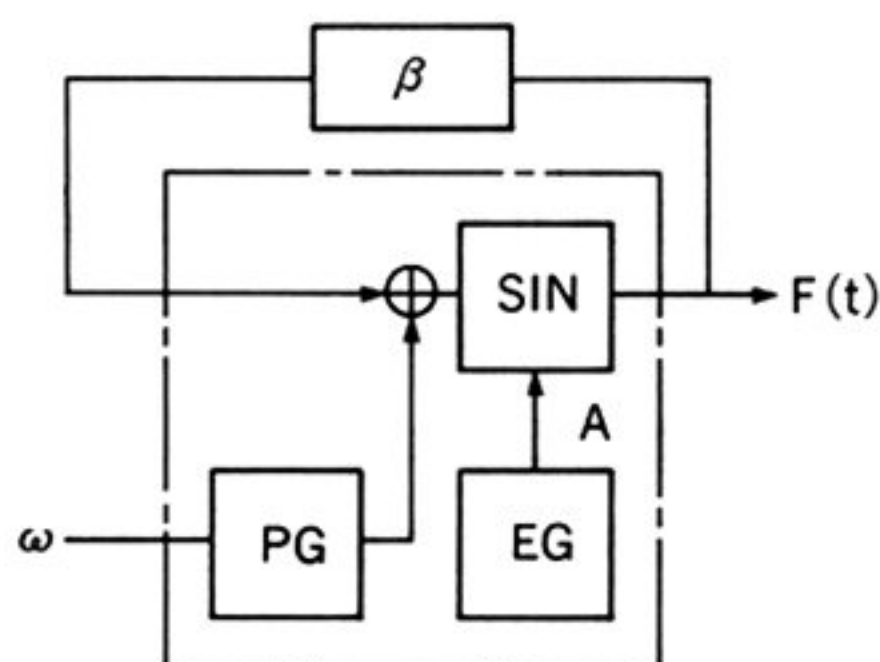
b. $F(t)=A_1 SIN \omega_1 t+A_2 SIN \omega_2 t$

c. $F(t)=A\ SIN(\omega t+\beta F(t))$

図7.37 ユニットセルによるFM方式の表現

4. ADPCM音声分析・合成の概略

MSX-AUDIOには、音声の分析・合成が簡単で音質も自然なADPCM方式と、分析はコンピュータを使った複雑な処理が必要であるが発音のためのメモリが比較的少なくてすむ複合正弦波音声合成方式の2種類の音声情報処理を備えています。ここでは、FM音源とともにこのLSIの柱であるADPCM方式による音声分析・合成について説明します。

ADPCM (Adaptive Differential Pulse Code Modulation) は、音声データと予測データとの差を、波形の推移によって柔軟に変化する量子化幅（適応量子化幅）によってコード化し、音質の低下を防ぐとともに、ビット数の削減を図った音声分析・合成方式です。以下で、そのコード化およびデコードの手順について述べます。

1. 音声分析

MSX-AUDIOでは8ビットのPCMデータを4ビットのADPCMデータへの変換を行っています。

1. まず、音声をサンプリングレート（1.8KHz〜16KHz）ごとに8ビットのPCMデータ（X^{1}_{n}）に変換します。

2. 得られたPCMデータ（X^{1}_{n}）を256倍し、16ビットデータ（X_n）として予測値（$\hat{X}_n$）と比較し、その差（dn）を求めます。予測値（$\hat{X}_n$）の初期設定値は8000Hです。

3. 差分（dn）が正のときは、ADPCMデータのMSB（L4）を「0」、負のときは「1」とし、同時に差分の絶対値（$|dn|$）を計算します。

4. 次に、この差分の絶対値（$|dn|$）と量子化幅（Δ_n）との関係が、表7.82のいずれに当たるかによって、ADPCMの残り3ビットを決定します。量子化幅（Δ_n）の初期設定値は7FHです。

表7.82　ADPCMコード表

L_4		L_3	L_2	L_1	条　件
dn≧0	dn＜0				
		0	0	0	$\lvert dn \rvert < \Delta n/4$
		0	0	1	$\Delta n/4 \leqq \lvert dn \rvert < \Delta n/2$
		0	1	0	$\Delta n/2 \leqq \lvert dn \rvert < \Delta n \times 3/4$
0	1	0	1	1	$\Delta n \times 3/4 \leqq \lvert dn \rvert < \Delta n$
		1	0	0	$\Delta n \leqq \lvert dn \rvert < \Delta n \times 5/4$
		1	0	1	$\Delta n \times 5/4 \leqq \lvert dn \rvert < \Delta n \times 3/2$
		1	1	0	$\Delta n \times 3/2 \leqq \lvert dn \rvert < \Delta n \times 7/4$
		1	1	1	$\Delta n \times 7/4 \leqq \lvert dn \rvert$

以上の操作で音声データからADPCMデータへのデータ変換は終わりです。

5. ADPCMのデータが得られると、次のステップの予測値（$\hat{X}_{n+1}$）と量子化幅（Δ_{n+1}）の更新を行います。

$$\hat{X}_{n+1} = (1-2\times L_4)\times(L_3+L_2/2+L_1/4+1/8)\times\Delta_n+\hat{X}_n$$

$$\Delta_{n+1} = F(L_3,\ L_2,\ L_1)\times\Delta_n$$

表 7.83 量子化幅変化率

L_3	L_2	L_1	f
0	0	0	0.9
0	0	1	0.9
0	1	0	0.9
0	1	1	0.9
1	0	0	1.2
1	0	1	1.6
1	1	0	2.0
1	1	1	2.4

以下、1〜5の操作を各サンプリングタイムごとに繰り返すことによって、完全なADPCM方式の音声分析が得られます。

2. 音声合成

1. 合成モードの手順は分析モードの5項で与えられる予測値・量子化幅の更新の式が、再生データを計算する式になります。つまり、予測値が再生音を与えることになります。しかし、これで得られる再生音は、サンプリング毎の階段波となり、ステップノイズの発生など音質にやや問題があります。そこで、MSX-AUDIOでは次の方法で波形をなめらかにします。

2. まず、5で得られる再生値を平均化回路に通します。このことは、このシステムに一種のLow Pass Filterを挿入したことと同じになり、広域の雑音をカットすることができます。さらに、ステップノイズを軽減するために各サンプリング間をさらに50KHzで再サンプリングし、線形補間します。この様子は図7.38のようになります。

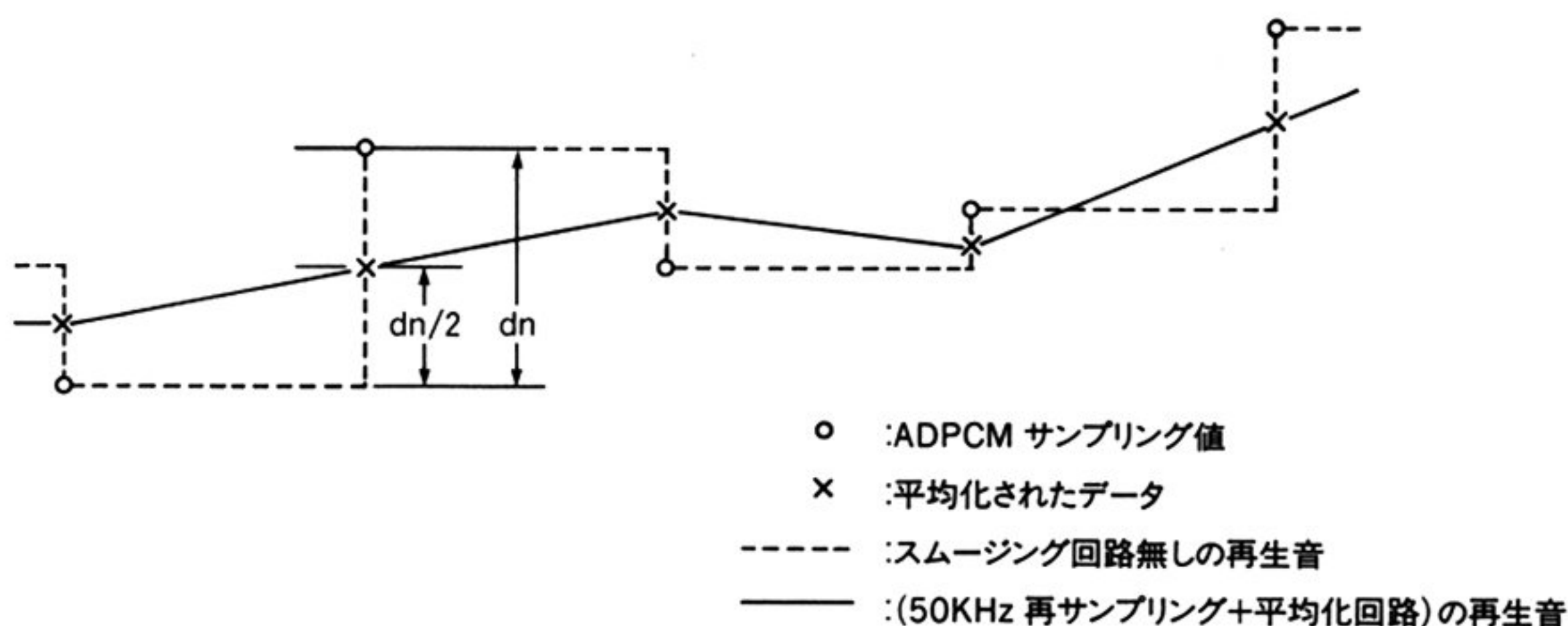



図 7.38 ADPCM 音声合成波形

4.5.2 機能概要

1. 主要機能

MSX-AUDIO の基本的構成は、FM 音源、ADPCM 方式音声分析・合成、外部メモリコントロール、AD/DA 変換および鍵盤スキャニングのための入出力ポートです。

1. FM 音源

FM 音源部の発音形態には、9 音同時発生モード、メロディ6 音＋リズム 5 音発生モードおよび複合正弦波音声合成モードの 3 種類があり、このいずれを使うかは、ソフトウェアで指定します。なお、この FM 音源部は OPL (YM3526) と同一であり、OPL のソフトウェアがそのまま使えます。

9 音同時発生モード

このモードでは FM 音 9 音を同時に、かつ異音色で発音できます。このとき、リズム選択ビット (R)、音声合成ビット (CSM) はともに「0」にセットしなければなりません。

メロディ6 音＋リズム 5 音モード

このモードではメロディ6 音とリズム 5 音を同時に、かつ異音色で発音できます。発音可能なリズム音は、バスドラム、スネアドラム、タムタム、トップシンバル、ハイハットシンバルの 5 種類です。

音声合成モード

この場合の音声合成は複合正弦波音声合成法を指します。音声を 3〜6 の sin 波によってシミュレートします。

2. ADPCM 方式音声分析・合成

4 ビットの ADPCM で音声の分析・合成を行います。サンプリングレートは、分析時は 1.8KHz〜16KHz、合成時は 1.8KHz〜50KHz の範囲で自由にプログラムできます。さらに、分析結果や合成データは、外付けメモリ (RAM、ROM)、CPU 側いずれでも記憶することができます。

3. 外部メモリコントロール

この機能は主として ADPCM で分析・合成される音声データを記憶するための外部メモリをコントロールします。外付け可能なメモリは 256Kbit DRAM、64Kbit DRAM、そしてバイト単位でアクセス可能な ROM です。容量は RAM、ROM ともに 256K バイトです。

4. AD/DA変換

ADPCM部のAD/DAの各変換器を単独で動作させることができます。ただし、このモードのときはFM音源およびADPCM音声分析・合成の動作は止まります。

5. 鍵盤スキャニング

ミュージックキーボードを外付け可能にするための入出力各8ビットのポートです。

以上の各機能の他に、より自然な音作りを可能にするビブラート発振器、振幅変調発振器や、各種の基準信号となる長短2つのタイマを備え、ソフトウェアの負担を軽くしています。また、汎用のI/Oポートが4ビット用意されています。

2. チャンネルとスロット

MSX-AUDIOはFM音を9音(9チャンネル)発音することが可能で、1音あたり2オペレータセル持っています。ただし、オペレータセルはシステムで1つ持っているだけなので、FM9音の計算は、このオペレータセルをシリアルに18回通すことによってなされます。このオペレータセルを通す順番(スロット番号)は、レジスタ番号と対応しており、各音の発音コントロールはスロットと対応したレジスタを制御することになります。

また、F-Numberのようなチャンネルごとのデータは2つのスロットを制御します。この2つのスロット(第1、第2スロット)の関係は、FM変調モードにした場合は、第1スロットが必ず変調波に、そして第2スロットが搬送波になります。また、第1スロットはFeedback FMのモードにも設定できます。このモード設定については「FEEDBACK/CONNECTION」を参照して下さい。

表7.84はチャンネルとスロットの関係を示します。

表7.84 チャンネルとスロット

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	スロット番号
1	2	3	1	2	3	4	5	6	4	5	6	7	8	9	7	8	9	チャンネル番号
1			2			1			2			1			2			チャンネルごとに見たときのスロット番号
20	21	22	20	21	22	23	24	25	23	24	25	26	27	28	26	27	28	チャンネルごとのデータとレジスタの関係(例 $20〜$28)
C0	C1	C2	C0	C1	C2	C3	C4	C5	C3	C4	C5	C6	C7	C8	C6	C7	C8	チャンネルごとのデータとレジスタの関係(例 $C0〜$C8)

3. レジスタマップ

アドレス	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
00〜1F	−CONTROL−							
20〜35	AM	VIB	EG-TYP	KSR	MULTI			
40〜55	KSL		TL					
60〜75	AR				DR			
80〜95	SL				RR			
A0〜A8	F−Number(L)							
B0〜B8			KON	Block			F-Num (H)	
BD	AM DEP	VIB DEP	R	BD	SD	TOM	TC	HH
C0〜C8					FB			C

アドレス	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	
00									
01	TEST								
02	TIMER 1								
03	TIMER 2								
04	IRQ RST	T1 MSK	T2 MSK	EOS MSK	BR MSK		ST2	ST1	
05	Key Board IN								*
06	Key Board OUT								
07	STA RT	REC	MEM DATA	REPT	SP OFF			RST	
08	CSM	NOTE SEL			Sam pl	DA AD	64K	ROM	
09	START ADD (L)								
0A	START ADD (H)								
0B	STOP ADD (L)								
0C	STOP ADD (H)								
0D	PRESCALE (L)								
0E	PRESCALE (H)								
0F	ADPCM−DATA								*
10	DELTA−N (L)								
11	DELTA−N (H)								
12	EG−CTRL								
15	DAC DATA H								
16	DAC DATA (L)								
17						SHIFT 2	1	0	
18					I/O−CTRL				
19					I/O DATA				*
1A	PCM−DATA								*

*READ可能なレジスタ

−STATUS−

INT	T1	T2	EOS	BUF RDY			PCM BSY

図7.39 MSX-AUDIOのレジスタマップ

4.5.3 動作説明

MSX-AUDIOは、プロセッサからレジスタアレイへデータを書き込むことによって制御されます。この書き込まれたデータによって、楽音のエンベロープの形状や変調度、周波数および発音モードなどが決定されます。そして、このデータの組み合せが、ピアノやバイオリンなどの音を発生させることになります。このマニュアルでは、各レジスタの機能を主として説明し、他のブロックについては簡単な説明に留め置きます。ただし、FM音源については、「4.5.4 楽音の作り方」の章で少し例を上げて説明します。

1. レジスタ

レジスタは図7.39のアドレスマップで与えられるように約1Kbitsのエリアを持っています。そして、この1Kbitsのエリアを機能毎にバイト単位でまとめ、各バイトにアドレスを割り当てます。

したがって、あるデータをMSX-AUDIO内に格納したい場合は、まずそのデータを格納するアドレスを指定し、次に目的のデータを送ります。ただし、同一サブアドレスを何度もアクセスする場合は、最初にアドレスを送るだけで、以後はアドレスデータを送ることなしに、データを送ることができます。

なお、初期設定のときに「0」にセットされるレジスタは、以下の各レジスタの説明の項で＊マークで示します。

TEST　$01

このアドレスはLSIの内部動作をテストするときに使用しており、「0」以外では正常動作しません。

■ TIMER

タイマは分解能80μsのタイマ1と分解能320μsのタイマ2の2種類あります。各タイマは、始動、停止およびフラグの制御が可能です。また、タイマのフラグが立ったときは、同時にIRQ端子はLOWレベルになり、タイマインタラプトをCPUに告げます。

TIMER-1　$02

タイマ1は8ビットのプリセッタブルカウンタによるタイマであり、このカウンタのオーバー

フローが生じたときに、タイマ1のフラグを立て、同時にプリセット値をロードします。

タイマ1は、通常のタイマ機能以外に複合音声合成のコントロールとしても働きます。この場合は、オーバーフローが生じたときに全スロットをKey ON(発音状態)にし、すぐさまKey OFFとします。この操作により複合音声合成を容易に実現することができます。

$02	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

$Tov(ms) = (256 - N_1) \times 0.08 \quad @\phi M = 3.6MHz$

$N_1 = D_7 \times 2^7 + D_6 \times 2^6 + \cdot \cdot \cdot + D_1 \times 2 + D_0$

TIMER-2 $03

タイマ2もタイマ1と同様に8ビットのプリセッタブルカウンタですが、タイマ1と異なる点は、タイマ1の分解能が80μsであるのに対して、タイマ2の分解能は320μsであることです。

$03	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0

$Tov(ms) = (256 - N_2) \times 0.32 \quad @\phi M = 3.6MHz$

$N_2 = D_7 \times 2^7 + D_6 \times 2^6 + \cdot \cdot \cdot + D_1 \times 2 + D_0$

FLAG CONTROL $04

このレジスタでは、タイマ1、タイマ2の始動、停止、フラグ制御およびADPCM、メモリコントロールのフラグの制御を行います。なお、初期設定時はD3、D4ビットのみ「1」にセットされ、他のビットは「0」にセットされます。

$04	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	IRQ RESET	MASK T1	MASK T2	MASK EDS	MASK BUF RDY		ST2	ST1

D_0 (ST1)　タイマ1の起動、停止を制御します。
このビットが「1」になったときに、タイマ1にプリセット値をロードしてカウントを始めます。このビットが「0」のときはタイマ1は動作しません。

D_1 (ST2)　タイマ2について D_0 (ST1) と同様の動作をします。

D_3 (MASK BUF RDY)
このビットは「1」のとき、CPUと音声分析・合成 (ADPCM)、あるいは外部メモリとデータのやり取りをしている場合のデータの書き込み要求、読み出し要求をマスクします。

D_4 (MASK EDS)
音声分析・合成 (ADPCM) あるいは外部メモリのREAD/WRITEの終了、さらにAD変換時の変換終了を示すフラグをマスクします。

D_5 (MASK T2)
このビットが「1」のときは、タイマ2のフラグは、タイマ2の動作に関係なく、常に「0」になります。

D_6 (MASK T1)
このビットはタイマ1のフラグをマスクします。

D_7 (IRQ RESET)
MSX-AUDIOの各フラグは該当するイベントが生じたときセット(「1」)され、$\overline{\mathrm{IRQ}}$は「0」になります。この状態を解除するために用意されているのがこのビットです。このビットが「1」になると、すべてのフラグは「0」になります。ただし、特定のフラグのみセットしたいときは、MASKビットに「1」を書き込んで下さい。

注　意	D_7ビットに「1」を書き込むと、フラグをリセットした後「0」にリセットされます。また、D_7が「1」のときはD_0〜D_6のデータは無視されます。

KEY BOARD IN　$05

IN_0〜IN_7の入力ポートを示すアドレスです。したがって、このアドレスは読み出し専用です。IN_0〜IN_7がデータバスD_0〜D_7に対応します。

$05	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	IN_7	IN_6	IN_5	IN_4	IN_3	IN_2	IN_1	IN_0

KEY BOARD OUT ＊ $06

OUT_0～OUT_7を示すアドレスです。このレジスタは「1」を書き込んだとき、電流をシンク（電圧0.4V以下）します。OUT_0～OUT_7がデータバスD_0～D_7に対応します。

$06	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	OUT_7	OUT_6	OUT_5	OUT_4	OUT_3	OUT_2	OUT_1	OUT_0

START/REC/MEM DATA/REPEAT/SP-OFF/RESET ＊ $07

ここでは、ADPCM音声分析・合成の起動、外部メモリアクセスの設定などの制御をします。

$07	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	START	REC	DATA MEMORY	REPEAT	SP-OFF			RESET

D_0（RESET）

外部メモリをデータ源としてADPCM音声合成を実行中に、このビットを「1」にすると、ADPCM合成回路および外部メモリコントローラは初期状態に戻ります。なお、この時REPEATビットは必ず「0」にしなくてはなりません。また、このREPEATはADPCM回路および外部メモリコントローラが暴走したときにも使えます。

D_3（SP-OFF）

このビットはSP-OFF端子と結ばれていて、D_3を「1」にするとSP-OFF端子は「0」

に、D_3を「0」にするとSP-OFF端子は「1」になります。ADPCM分析時、あるいはAD変換時にスピーカーを保護するコントロールとして使います。

D_4 (REPEAT)

外部メモリを使ってADPCM音声合成を行っている場合、同一区間(スタートアドレスからストップアドレスまで)を何度も繰り返して合成したいときに「1」にします。

D_5 (MEMORY DATA)

外部メモリをアクセスするとき、このビットを「1」にします。

D_6 (REC)　ADPCM音声分析あるいはCPUから外部メモリにデータを書き込むときに「1」にします。

D_7 (START)

ADPCM音声分析・合成を行うとき「1」にします。この場合、データ格納場所(CPUあるいは外部メモリ)によって開始タイミングが異なります。データ格納場所がCPUのときは、アドレス$0FをREAD/WRITEしたときから始まり、外部メモリのときはSTARTビットが「1」になったときに始まります。したがって、外部メモリアクセスのときは、STARTビットを「1」にする前に、他のすべての条件を整えておかなければなりません。また、STARTビットを「0」にするときは、先にSTARTビットを「0」にして、残りのデータをリセットします。

CSM/KEY BOARD SPLIT/SAMPLE/DA AD/64K/ROM ＊ $08

このアドレスでは、複合正弦波音声合成モード、AD/DA変換、外部メモリのタイプ指定を制御します。

$08	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	CSM	NOTE SEL			SAMPL	DA/AD	64K	ROM

D_0 (ROM)　外部メモリのタイプ指定です。「0」= RAM、「1」= ROM。

D_1 (64K)　外部メモリのタイプ指定です。「0」= 256Kbit DRAM、「1」= 64Kbit DRAM。このビットが「1」のとき、A8 アドレスの出力は意味がありません。ROM は「0」です。

D_2 (DA/AD)

このビットは次の SAMPL と組み合わせて使います。「1」のときは、MO 出力は$15 ～$17 で指定されるデータを出力し、「0」のときは SAMPLE が「1」であれば、AD 変換、「0」だと MUSIC を出力します。

D_3 (SAMPLE)

AD 変換、DA 変換時のタイマをイネーブルにするビットです。AD 変換はこのビットを「1」にしたときから始まります。

D_6 (NOTE SEL)

このビットは、1 オクターブ内の鍵盤分割の分離点を制御します。「0」のときはキーボードスプリットを指示するビットを F-Number の MSB から見て 2 番目のビットで、「1」のときは、F-Number の MSB でコントロールします。この様子は次の表のとおりです。F-Number / BLOCK の項を参照して下さい。

D_6=「0」

0		1		2		3		4		5		6		7		オクターブ
0		1		2		3		4		5		6		7		BLOCK データ
1		1		1		1		1		1		1		1		F-Number MSB
0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	F-Number 2nd
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	キーボードスプリット番号

D_6=「1」

0		1		2		3		4		5		6		7		オクターブ
0		1		2		3		4		5		6		7		BLOCK データ
1		1		1		1		1		1		1		1		F-Number MSB
*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*	F-Number 2nd
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	キーボードスプリット番号

*定めず

D_7 (CSM)　「1」が設定されると、複合正弦波音声合成モードになります。このときは、全チャンネルともKey-OFF状態にしておかなければなりません。

START ADDRESS L/H　$09 $0A

外部メモリをアクセス (ADPCM回路/CPU) するときの、メモリのスタート番地をL ($09)、H ($0A) の16ビットで与えます。ただし、ROMとRAMでは指定の方法が少し異なります。

$09

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
START ADDRESS (L)							

$0A

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
START ADDRESS (H)							

■ RAMの場合

1. 64Kbit DRAM

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0A−								−$09−												
D_7	D_6	D_5	*	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	*	D_2	D_1	D_0	0	0	0	0	0

*$0Aの$D_4$と$09のD_3は必ず「0」でなくてはならない。

2. 256Kbit DRAM

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0A−								−$09−												
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	0	0	0	0	0

■ ROMの場合

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0A−								−$09−												
*	*	*	D_4	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	0	0	0	0	0

*$0Aの$D_5$〜$D_7$は$0Cのデータと同じでなくてはならない。

STOP ADDRESS L/H　$0B $0C

外部メモリをアクセス(ADPCM回路/CPU)するときの、メモリのストップ番地をL($0B)、H ($0C) の16ビットで与えます。ただし、ROMとRAMでは指定の方法が少し異なります。

$0B	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	STOP ADDRESS (L)							

$0C	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	STOP ADDRESS (H)							

■ RAMの場合

1. 64Kbit DRAM

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0C−								−$0B−												
D_7	D_6	D_5	*	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	*	D_2	D_1	D_0	|	|	|	|	|

*$0Cの$D_4$と$0BのD_3は必ず「0」でなくてはならない。

2. 256Kbit DRAM

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0C−								−$0B−												
D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	|	|	|	|	|

■ ROMの場合

BANK			CAS ADDRESS									RAS ADDRESS								
2^2	2^1	2^0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
−$0C−								−$0B−												
*	*	*	D_4	D_3	D_2	D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	|	|	|	|	|

*$0Cの$D_5$〜$D_7$は$0Aのデータと同じでなくてはならない。

PRESCALE L/H $0D $0E

AD変換時(ADPCM分析を含む)のサンプリングレートおよびDA変換時のサンプリングレートを指定します。サンプリングレートは次の式で与えられ、最大サンプリングレートは16KHz、最小は1.8KHzです。

$$fsample = 3.6MHz / N_{PRE}$$

N_{PRE}はプリスケール値　$225 \leqq N_{PRE} \geqq 2047$

式7.11 サンプリングレート

$0D

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
PRESCALE (L)							

$0E

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
					PRESCALE (H)		

ADPCM-DATA $0F

ADPCM分析・合成をCPUと行うときは、このレジスタを介してデータのやりとりをします。また、CPUから外部メモリをアクセスするときも同様にこのレジスタをバッファとして使います。

$0F

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
ADPCM-DATA							

注　意　ADPCM分析・合成のデータ構成

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
n番目データ				n+1番目データ			
L_4	L_3	L_2	L_1	L_4	L_3	L_2	L_1

ADPCMのデータは左記のように、1バイトで2データを構成しており、上位4ビットがn番目のデータとすると、下位4ビットはそれに続くn+1番目のデータとなります。

DELTA-N L/H　$10 $11

ADPCM音声合成時に各サンプリング間を50KHzで線形補間するための補間係数を与えます。また、このデータは合成時のサンプリングレートを指定することになります。したがって、合成時はプリスケールデータを使用しません。

$$\Delta N = k \times 2^{16},\ k = \left(\frac{3.6\text{MHz}}{50\text{KHz}}\right) \ / \ \left(\frac{3.6\text{MHz}}{\text{fsample}}\right)$$

$$\text{VOICE}_{n,i} = \text{VOICE}_n + (\text{Noff}_n + i_n \times k) \times (\text{VOICE}_{n+1} - \text{VOICE}_n)$$

$$\begin{cases} 0 \leqq \text{Noff}_n + i_n \times k < 1 \\ \text{Noff}_n < k,\ \text{Noff}_n = \text{Noff}_{n-1} + i'_{n-1} \times k + k - 1 \end{cases}$$

i'_{n-1}はn-1回目の最大値

$10

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DELTA-N (L)							

$11

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
DELTA-N (H)							

ENVELOP CONTROL (at ADPCM)　$12

ADPCM音声合成の出力レベルを無音から最大音256段階にボリュームコントロールをします。このデータはADPCM音声合成出力のみに有効で、MUSIC出力やDA変換に対しては無効です。

$$\text{AUDIO OUT} = \text{VOICE}_n \times \text{EG}$$

$12	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
	EG-CONTROL							

DAC-DATA $15 ~ $17

DA 変換を行うときのデジタルデータを与えます。この3バイト13ビットのデータで以下の式で与えられるアナログ値を出力（DACを経由して）します。この場合、アドレス$15にデータを書き込んだときがトリガとなり、$15～$17のレジスタの内容を出力します。なお、DA 変換を行うときは、レジスタ$08の D_2（DA／AD）を「1」にする前に、必ず初期値を$15～$17のレジスタに書いて下さい。

$$\begin{cases} V_{OUT}=\dfrac{Vcc}{2}+\dfrac{Vcc}{4}\times(-1+F_9+F_8\times2^{-1}+\cdots\cdots\cdots+F_1\times2^{-8}+F_0\times2^{-9}+2^{-10})\times2^{-E} \\ E=\overline{S}_2\times2^2+\overline{S}_1\times2^1+\overline{S}_0\times2^0 \qquad @ \quad S_0+S_1+S_2\geqq1 \end{cases}$$

	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
$15	F_9	F_8	F_7	F_6	F_5	F_4	F_3	F_2
$16	F_1	F_0						
$17						S_2	S_1	S_0

I/O-CONTROL AND I/O-DATA $18 $19

MSX-AUDIOでは汎用I/Oポートを4ビット用意しており、このI/Oポートをコントロールするレジスタがアドレス$18と$19です。

$18はI/Oポートの入出力の方向制御ビットで、「1」のとき、ポートは出力に、「0」のときは入力になります。初期状態では「0」です。

$19はI/Oポートのデータをやりとりするレジスタです。

$18	D7 D6 D5 D4	D3	D2	D1	D0
		IO3	IO2	IO1	IO0
		I/O3 CTRL	I/O2 CTRL	I/O1 CTRL	I/O0 CRTL

$19	D7 D6 D5 D4	IO3	IO2	IO1	IO0
		I/O3 DATE	I/O2 DATE	I/O1 DATE	I/O0 DATE

PCM-DATA　$1A

AD変換実行時に、変換済みデータを格納するレジスタです。

$1A	D7 D6 D5 D4 D3 D2 D1 D0
	PCM-DATA

注 意　PCMコードは2の補数です。

AM/VIB/EG-TYP/KSR/MULTIPLE *　$20～$35

このレジスタでは、エンベロープの形状やF-Numberで与えられる周波数データを楽音の周波数成分に見合った搬送波、変調波の周波数に変換するための倍率を制御します。

$20～$35	D7	D6	D5	D4	D3	D2	D1	D0
	AM	VIB	EG-TYP	KSR	MULTI			
					2^3	2^2	2^1	2^0

D_0〜D_3 (MULTIPLE)

表7.85で与えられる倍率によって搬送波、変調波の周波数を制御します。

例

F-Numberによる周波数	ωf
搬送波のMULTIPLE	1
変調波のMULTIPLE	7

$$F(t)=E\sin(\omega ft+I\sin(7\omega ft))$$

表7.85 倍率

MUL	倍率	MUL	倍率	MUL	倍率	MUL	倍率
0	1/2	4	4	8	8	C	12
1	1	5	5	9	9	D	12
2	2	6	6	A	10	E	15
3	3	7	7	B	10	F	15

D_4 (KSR) RATEのキースケールを与えます。自然楽器では、おおむね音程が高くなるにしたがって、音の立ち上がり、立ち下がりは速くなります。この現象をシミュレートするのがRATEのキースケールであり、表7.86の値が各々の音程に対してスピードのオフセットとして加えられます。したがって、実際のRATEはADSRに対して設定したRATEにこのオフセットを加えたものになります。

$$RATE=4\times R+Rks$$

Rは ADSRでの設定値
Rksはキースケールオフセット値
ただし、R=0のときはRATE=0

表 7.86 RATE のキースケール

D_4	N	Rks	N	Rks	N	Rks	N	Rks
	0	0	4	1	8	2	12	3
	1	0	5	1	9	2	13	3
0	2	0	6	1	10	2	14	3
	3	0	7	1	11	2	15	3
	0	0	4	4	8	8	12	12
	1	1	5	5	9	9	13	13
1	2	2	6	6	10	10	14	14
	3	3	7	7	11	11	15	15

D_5 (EG-TYP)

持続音か減衰音かの切り換えをします。

D_5=「0」のとき、減衰音

D_5=「1」のとき、持続音

この発音モードの違いは、RELEASE RATE の使用法が異なっているためで、その様子を図 7.40 に示します。

D_5= "0" 減衰音　　D_5= "1" 持続音

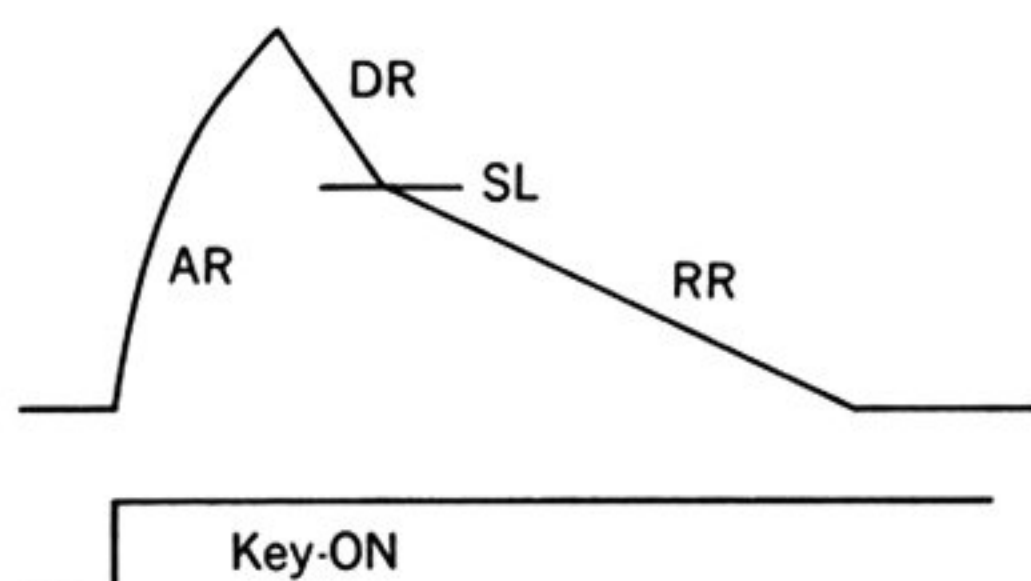
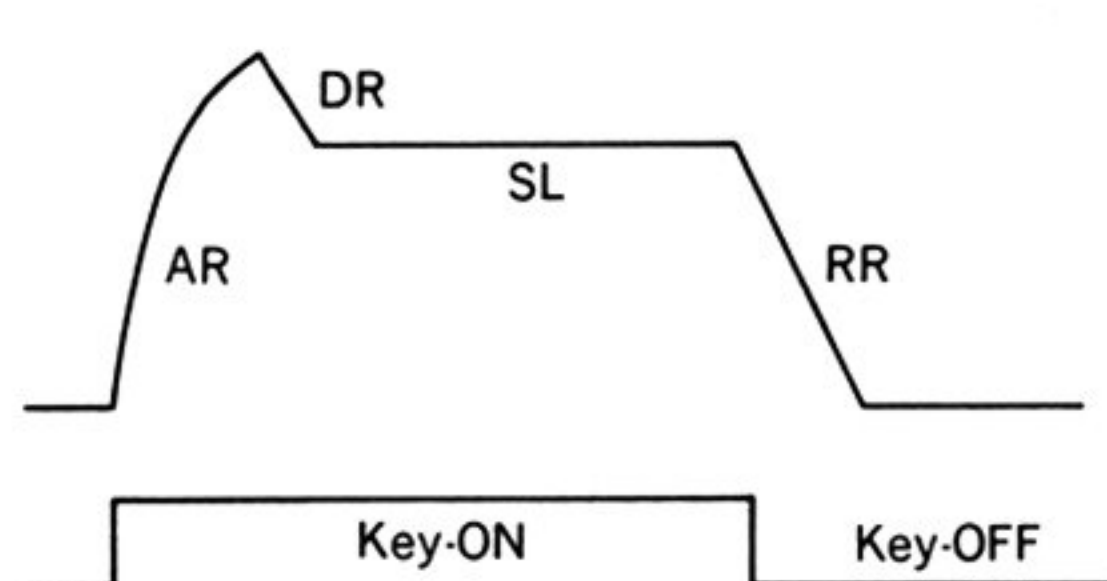
DR SL AR RR Key-ON　　DR SL AR RR Key-ON Key-OFF

AR=ATTACK RATE　DR=DECAY RATE　SL=SUSTAIN LEVEL
RR=RELEASE RATE

図 7.40 2種類の発音モード

D_6 (VIB)　ビブラートの ON / OFF スイッチです。このビットを「1」にすると、そのスロットにはビブラートがかかります。このときの周波数は 6.4Hz (@ϕM = 3.6MHz) で、ビブラートの深さは、BD レジスタの VIB-DEPTH で決まります。

D_7 (AM)　振幅変調の ON / OFF スイッチです。このビットが「1」にセットされたときには、そのスロットには振幅変調がかかります。振幅変調の周波数は 3.7Hz (@ϕM = 3.6MHz) です。

KSL/TOTAL LEVEL ＊ $40〜$55

トータルレベルとは、エンベロープジェネレータの出力に対して、減衰量を加算し変調度（音色）および音量の制御をするために用いられます。また、レベルキースケール(KSL)は、RATEのキースケール同様、自然楽器では音程が上がるにつれて、出力レベルは低下する傾向にあることをシミュレートするものです。

$40〜$45

D_7 D_6	D_5 D_4 D_3 D_2 D_1 D_0
KSL	Total Level

D_0〜D_5（Total Level）

減衰量の最小分解能は0.75dBで、最大47.25dBまで音量を絞り込むことができる。

表7.87　トータルレベル

	D_5	D_4	D_3	D_2	D_1	D_0
減衰量	24dB	12dB	6dB	3dB	1.5dB	0.75dB

D_6、D_7（KSL）

レベルのキースケールを調整するビットです。キースケールのモードは、音程が上がるほどレベルは減衰し、その減衰量は、1.5dB/OCT、3dB/OCT、6dB/OCT、および減衰量なしの4種類です。

表7.88　キースケールの減衰量

D_6	D_7	減衰量
0	0	0
1	0	1.5dB/OCT
0	1	3dB/OCT
1	1	6dB/OCT

表 7.89　3dB/OCT の場合の各 F-Number での減衰量

F-Number	0	1	2	3	4	5	6	7
OCT	8	9	10	11	12	13	14	15
0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
	0.000	0.075	1.125	1.500	1.875	2.250	2.625	3.000
2	0.000	0.000	0.000	0.000	0.000	1.125	1.875	2.625
	3.000	3.750	4.125	4.500	4.875	5.250	5.625	6.000
3	0.000	0.000	0.000	1.875	3.000	4.125	4.875	5.625
	6.000	6.750	7.125	7.500	7.875	8.250	8.625	9.000
4	0.000	0.000	3.000	4.875	6.000	7.125	7.875	8.625
	9.000	9.750	10.125	10.500	10.875	11.250	11.625	12.000
5	0.000	3.000	6.000	7.875	9.000	10.125	10.875	11.625
	12.000	12.750	13.125	13.500	13.875	14.250	14.625	15.000
6	0.000	6.000	9.000	10.875	12.000	13.125	13.875	14.625
	15.000	15.750	16.125	16.500	16.875	17.250	17.625	18.000
7	0.000	9.000	12.000	13.875	15.000	16.125	16.875	17.625
	18.000	18.750	19.125	19.500	19.875	20.250	20.625	21.000

単位(dB)

注　意　F-Number は上位 4 ビットの値
1.5dB/OCT は上記の 1/2 倍
6dB/OCT は上記の 2 倍

ATTACK/DECAY RATE ＊　$60 ～ $75

アタックレイトは音の立ち上がり時間の設定をします。また、ディケイレイトは、アタック後の減衰時間を決めます。各 RATE の時間設定は表 7.90 のとおりです。

$60～$75

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
AR				DR			
2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0

SUSTAIN LEVEL/RELEASE RATE　$80 ～ $95

サスティンレベルは、持続音の場合は、ディケイモードでの減衰がこのレベルに到達するとそ

の後はそのレベルを保持するという変化点を指し、減衰音の場合は、ディケイモードからリリースモードへの変化点を与えます。

リリースレベルは、持続音の場合はKeyをOFFしたときに、音が消えてゆく様子を定義するRATEであり、減衰音のときはサスティンレベルの前の減衰をディケイレイトで表し、サスティンレベル後の減衰をこのリリースレイトで表します。

$80〜$95

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
SL				RR			
24 dB	12 dB	6 dB	3 dB	2^3	2^2	2^1	2^0

注 意 D_4〜D_7（SL）がすべて「1」のときは、93dBとする。
リリースレイトの減衰時間は、ディケイ・レイトの表と同じ。

下記のレイトは、キースケール後のレイトです。また、レイトの値を上位4ビット(RM)と下位2ビット（RL）に分割してRM-RLと表しています。RATE = RM×4+RL

表7.90 各レイトでの立ち上がり、立ち下がり時間

EG ATTACK TIME RATE	[mS] (@10%−90%)	EG DECAY TIME RATE	[mS]	EG ATTACK TIME RATE	[mS] (@0dB−96dB)	EG DECAY TIME RATE	[mS]
15 3	0.00	15 3	0.51	15 3	0.00	15 3	2.40
15 2	0.00	15 2	0.51	15 2	0.00	15 2	2.40
15 1	0.00	15 1	0.51	15 1	0.00	15 1	2.40
15 0	0.00	15 0	0.51	15 0	0.00	15 0	2.40
14 3	0.11	14 3	0.58	14 3	0.20	14 3	2.74
14 2	0.11	14 2	0.69	14 2	0.24	14 2	3.20
14 1	0.14	14 1	0.81	14 1	0.30	14 1	3.84
14 0	0.19	14 0	1.01	14 0	0.38	14 0	4.80
13 3	0.22	13 3	1.15	13 3	0.42	13 3	5.48
13 2	0.26	13 2	1.35	13 2	0.46	13 2	6.40
13 1	0.31	13 1	1.62	13 1	0.56	13 1	7.68
13 0	0.37	13 0	2.02	13 0	0.70	13 0	9.60
12 3	0.43	12 3	2.32	12 3	0.80	12 3	10.96
12 2	0.49	12 2	2.68	12 2	0.92	12 2	12.80
12 1	0.61	12 1	3.22	12 1	1.12	12 1	15.36
12 0	0.73	12 0	4.02	12 0	1.40	12 0	19.20
11 3	0.85	11 3	4.62	11 3	1.56	11 3	21.92
11 2	0.97	11 2	5.38	11 2	1.84	11 2	25.56

EG ATTACK TIME RATE	[mS] (@10%−90%)	EG DECAY TIME RATE	[mS]	EG ATTACK TIME RATE	[mS] (@0dB−96dB)	EG DECAY TIME RATE	[mS]
11 1	1.13	11 1	6.42	11 1	2.20	11 1	30.68
11 0	1.45	11 0	8.02	11 0	2.76	11 0	38.36
10 3	1.70	10 3	9.24	10 3	3.12	10 3	43.84
10 2	1.94	10 2	10.76	10 2	3.68	10 2	51.12
10 1	2.26	10 1	12.84	10 1	4.40	10 1	61.36
10 0	2.90	10 0	16.04	10 0	5.52	10 0	76.72
9 3	3.39	9 3	18.48	9 3	6.24	9 3	87.68
9 2	3.87	9 2	21.52	9 2	7.36	9 2	102.24
9 1	4.51	9 1	25.68	9 1	8.80	9 1	122.72
9 0	5.79	9 0	32.08	9 0	11.04	9 0	153.44
8 3	6.78	8 3	36.96	8 3	12.48	8 3	175.36
8 2	7.74	8 2	43.04	8 2	14.72	8 2	204.48
8 1	9.02	8 1	51.36	8 1	17.60	8 1	245.44
8 0	11.58	8 0	64.16	8 0	22.08	8 0	306.88
7 3	13.57	7 3	73.92	7 3	24.96	7 3	350.72
7 2	15.49	7 2	86.08	7 2	29.44	7 2	408.96
7 1	18.05	7 1	102.72	7 1	35.20	7 1	490.88
7 0	23.17	7 0	128.32	7 0	44.16	7 0	613.76
6 3	27.14	6 3	147.84	6 3	49.92	6 3	701.44
6 2	30.98	6 2	172.16	6 2	58.88	6 2	817.92
6 1	36.10	6 1	205.44	6 1	70.40	6 1	981.76
6 0	46.34	6 0	256.64	6 0	88.32	6 0	1227.52
5 3	54.27	5 3	295.68	5 3	99.84	5 3	1402.88
5 2	61.95	5 2	344.32	5 2	117.76	5 2	1635.84
5 1	72.19	5 1	410.88	5 1	140.80	5 1	1963.52
5 0	92.67	5 0	513.28	5 0	176.64	5 0	2455.04
4 3	108.54	4 3	591.36	4 3	193.68	4 3	2805.76
4 2	123.90	4 2	688.64	4 2	235.52	4 2	3271.68
4 1	144.38	4 1	821.76	4 1	281.60	4 1	3927.04
4 0	185.34	4 0	1026.56	4 0	353.28	4 0	4910.08
3 3	217.09	3 3	1182.72	3 3	399.36	3 3	5611.52
3 2	247.81	3 2	1377.28	3 2	471.04	3 2	6543.36
3 1	288.77	3 1	1643.52	3 1	563.20	3 1	7854.08
3 0	370.69	3 0	2053.12	3 0	706.56	3 0	9820.16
2 3	434.18	2 3	2365.44	2 3	798.72	2 3	11223.04
2 2	495.62	2 2	2754.56	2 2	942.08	2 2	13086.72
2 1	577.54	2 1	3287.04	2 1	1126.40	2 1	15708.16
2 0	741.38	2 0	4106.24	2 0	1413.12	2 0	19640.32
1 3	868.35	1 3	4730.88	1 3	1597.44	1 3	22446.08
1 2	991.23	1 2	5509.12	1 2	1884.16	1 2	26173.44
1 1	1155.07	1 1	6574.08	1 1	2252.80	1 1	31416.32
1 0	1482.75	1 0	8212.48	1 0	2826.24	1 0	39280.64

注 意　レイトが「0」の場合は、エンベロープは変化しません。

BLOCK/F-Number/SUS/KEY ＊ 　$A0 ～ $B8

音程、音階を決めるデータです。F-Number は$A0～$A8 のレジスタと$B0～$B8 のレジスタにまたがっています。

$A0～$A8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
F-Number							
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

$B0～$B8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
		Key-ON	BLOCK			F-Number	
			2^2	2^1	2^0	2^9	2^8

D_0～D_7 （$20～$28）、D_0 （$B0～$B8） （F-Number）

$A0～$A8 の 8 ビットと$B0～$B8 の下位 1 ビットの計 9 ビットで F-Number を表します。この F-Number は音階を与えるデータで、後述する方法でその値を決めます。

D_2～D_4 （BLOCK）

オクターブ情報を与えます。

D_5 （KEY ON）

鍵盤の ON / OFF に相当するビットです。このビットを「1」にすると、そのチャンネルが ON となり、発音します。「0」で KEY OFF です。

■ F-Number / BLOCK

MSX-AUDIO では、必要な周波数はその周波数に応じた位相の増分を与えることにより得ることができます。そして、この増分は F-Number と BLOCK および MULTIPLE 情報によって決められます。

そこで、まず希望周波数の増分を求めます。これは次式で与えられます。

$\Delta P = fmus \times 2^{19} / fsam$　　　　$fsam = fM / 72$

fmus ： 希望周波数
fsam ： サンプリング周波数 (50KHz)
fM　： 入力クロック周波数 (3.6MHz)

式 7.11 希望周波数の増分

これで位相の増分は求めれらますが、この値を管理するのはビット数が多くて大変なため、増分は1オクターブ分のデータのみとし、各オクターブに対してはその増分をシフト(2倍、4倍・・・)することによって求めます。これにより増分は次のように表現できます。

$\Delta P = 2^{B} \times F' \times MUL$

B　： オクターブ情報
F'　： 1オクターブ内に制限した増分
MUL ： MULTIPLEデータ

式 7.12 シフトによる位相の増分

7.11、7.12式と増分 (F') を10ビットで表すということから、F-NumberとBLOCKは次のように表現されます。

$F = (fmus \times 2^{19} / fsam) / 2^{b-1}$　　　(@MUL = 1)

F ： F-Numberデータ
b ： BLOCKデータ

式 7.13 F-Number と BLOCK

表 7.91 F-Number (その1)

音階	周波数 (4oct)	F-Number	$B0~$B8		$A0~$A8							
			D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
C#	277.2	363	0	1	0	1	1	0	1	0	1	1
D	293.7	385	0	1	1	0	0	0	0	0	0	1
D#	311.1	408	0	1	1	0	0	1	1	0	0	0
E	329.6	432	0	1	1	0	1	1	0	0	0	0
F	349.2	458	0	1	1	1	0	0	1	0	1	0
F#	370.0	485	0	1	1	1	1	0	0	1	0	1
G	392.0	514	1	0	0	0	0	0	0	0	1	0
G#	415.3	544	1	0	0	0	1	0	0	0	0	0
A	440.0	577	1	0	0	1	0	0	0	0	0	1
A#	466.2	611	1	0	0	1	1	0	0	0	1	1
B	493.9	647	1	0	1	0	0	0	0	1	1	1
C	523.3	686	1	0	1	0	1	0	1	1	1	0

表 7.92 F-Number (その2)

音階	周波数 (4oct)	F-Number	$B0~$B8		$A0~$A8							
			D_1	D_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
G	392.0	514	1	0	0	0	0	0	0	0	1	0
G#	415.3	544	1	0	0	0	1	0	0	0	0	0
A	440.0	577	1	0	0	1	0	0	0	0	0	1
A#	466.2	611	1	0	0	1	1	0	0	0	1	1
B	493.9	647	1	0	1	0	0	0	0	1	1	1
C	523.3	686	1	0	1	0	1	0	1	1	1	0
C#	554.4	727	1	0	1	1	0	1	0	1	1	1
D	587.3	770	1	1	0	0	0	0	0	0	1	0
D#	622.2	816	1	1	0	0	1	1	0	0	0	0
E	659.3	864	1	1	0	1	1	0	0	0	0	0
F	698.5	916	1	1	1	0	0	1	0	1	0	0
F#	740.0	970	1	1	1	1	0	0	1	0	1	0

AM・VIB-DEPTH/RHYTHM ＊ $BD

振幅変調 (AM)、ビブラート (VIB) の深さおよびリズムのモード選択と各リズム楽器の ON / OFF をコントロールします。

$BD

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
AM-DEPTH	VIB-DEPTH	RHYTHM	BD	SD	TOM	TOP-CY	HH

D_0〜D_5 (RHYTHM)

D_5=「1」のとき、MSX-AUDIOはリズム音モードになり、7〜9チャンネルはリズム音のチャンネルとなります。したがって、楽音(メロディ部)は6音に制限されます。D0〜D4は各リズム楽器のON/OFFを制御します。このため$B6、$B7、$B8のKey ONビットは常に「0」にしておく必要があります。また、13〜18の各スロットはリズム音と表7.93のような対応をしており、RATEなどのデータは各リズム音にマッチした値を入力しなければなりません。

表7.93 リズムスロットと周波数データ

楽　器	スロット
DB	13、16
SD	17
TOM	15
TOP-CYM	18
HH	14

D_6 (VIB-DEPTH)

ビブラートの深さは2種類あり、D_6=「1」のとき14セント、D_6=「0」のときは7セントです。

D_7 (AM-DEPTH)

振幅変調の深さも2種類あります。

D_7=「1」のとき4.8dB

D_7=「0」のとき1dB

FEEDBACK/CONNECTION ＊ $C0 ~ $C8

このレジスタはSelf-Feedbackの変調度およびFM変調のタイプを決めます。

$C0~$C8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
				FEEDBACK 2^2	2^1	2^0	CONNECTION

D_0 (CONNECTION)

コネクションは2つのスロットの結線を制御します。データ「0」でFM変調モードとなり、データ「1」で2つのスロットが並列でサイン波の合成モードになります。

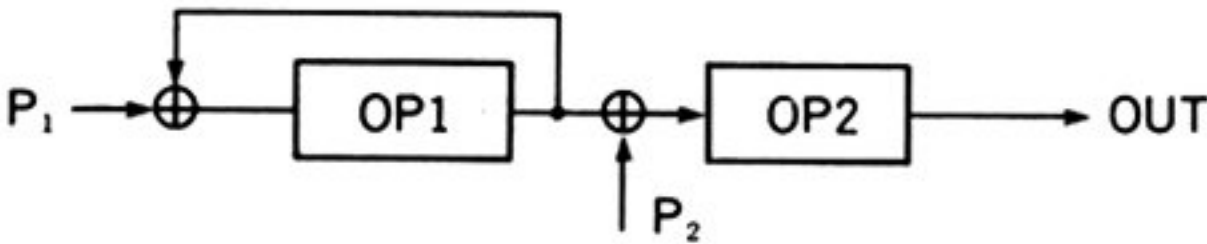
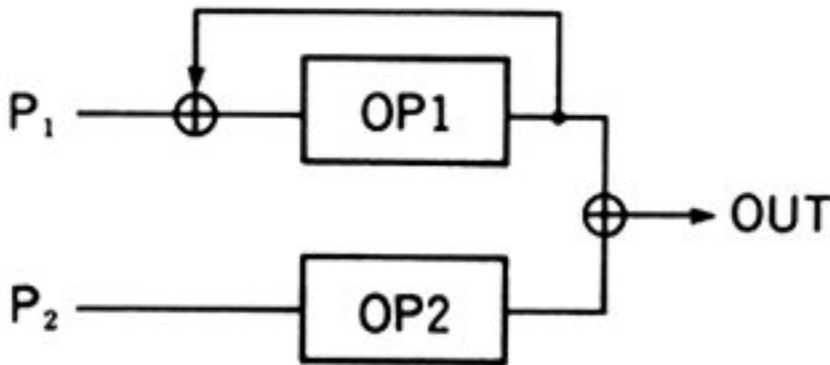



図7.41 CONNECTION

D_1~D_3 (FEEDBACK)

第1スロットのフィードバックFM変調の変調度を与えます。

表7.94 変調度

	0	1	2	3	4	5	6	7
変調度	0	$\pi/16$	$\pi/8$	$\pi/4$	$\pi/2$	π	2π	4π

2. フェイズジェネレータ（PG）

フェイズジェネレータは必要な周波数に応じた増分を単位時間ごとにアキュムレートして位相値を得る回路です。この増分はレジスタから送られてくる周波数情報（F-Number、BLOCK、MULTIPLE)から作られます。さらに、ビブラート発振器を内蔵しているため、この発振器の出力と周波数情報とを組み合わせることにより、ビブラート効果を作り出します。

3. エンベロープジェネレータ（EG）

エンベロープジェネレータ（以下、EG）は、ATTACK、DECAY、RELEASEの各RATE、Sustain Level、Total Levelなどでコントロールされ、音色、音量の経時変化を与えます。そして、そのダイナミックレンジは96dB (分解能0.1875dB) あります。EGは対数表示であり、また減衰量で表されます。その一般的な波形は図7.42のとおりです。この波形で特徴的なのは、アタック時は指数関数的に変化し、それ以外では直線的に変化する点です。また、アタックからディケイへの切り換えは、0dBに達したときに起こり、ディケイからサスティンへは、サスティンレベルに到達したときに起こります。そして、リリースへの移行はKeyがOFFされたときに起こります。トータルレベル、レベルキースケール、振幅変調などの効果は、その設定値をEGに加えることによりエンベロープの波形を変化させます。

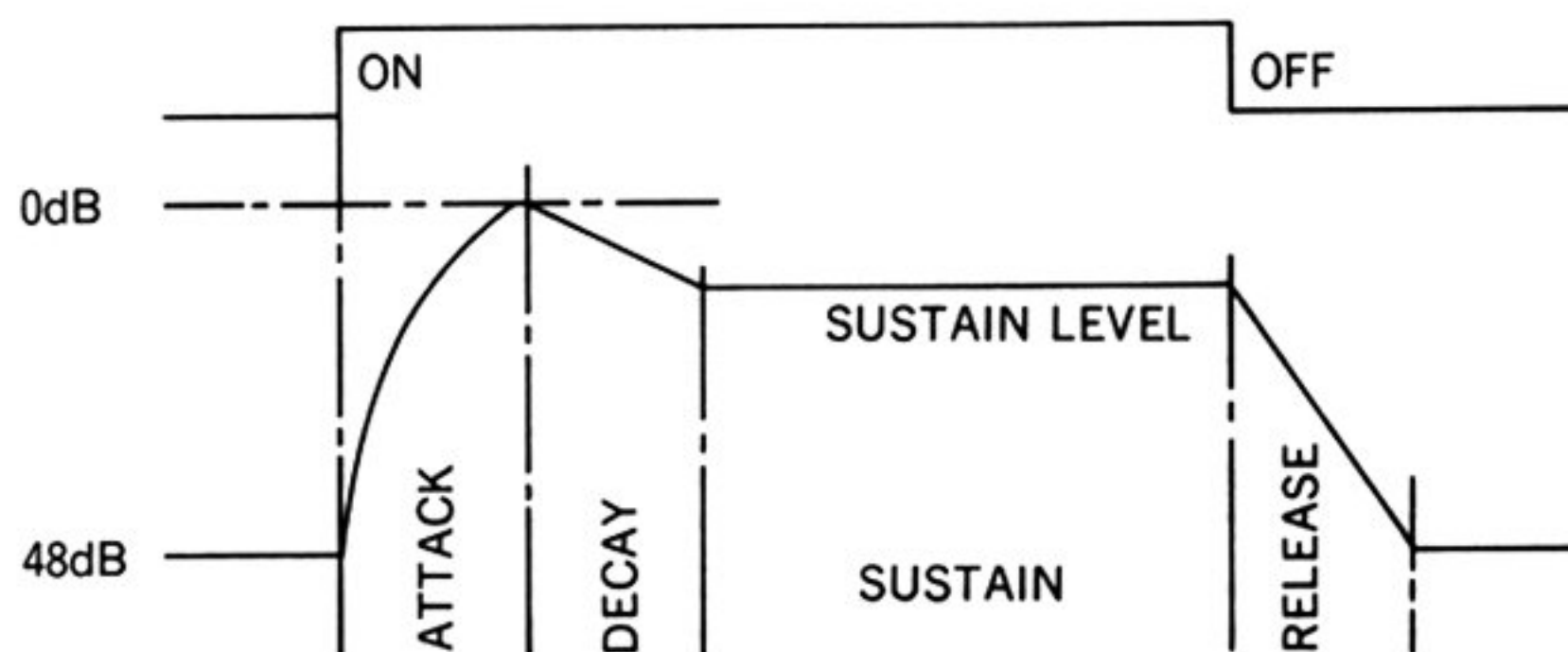



図7.42　エンベロープ波形

4. オペレータ（OP）とアキュムレータ（ACC）

オペレータはFM演算を行う回路です。オペレータでは、フェイズジェネレータからの位相出力をもとに、SINの値を計算し、これにエンベロープジェネレータ出力を掛け合わせます。この結果を変調波であればオペレータの入力へ返し、楽音であればアキュムレータへ送ります。この転送を制御するのがFeedback、CONNECTIONの各データです。

アキュムレータは各チャンネルのオペレータ出力を累積加算します。さらに、この加算結果を仮数部10ビット（サインビットを含む)、指数部3ビットのオフセットバイナリへデータ変換を行い、LSBより図7.43のように出力します。

ϕSY

SH

MO

DON'T CARE	LSB	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	SIGN	S_1	S_2	S_3	DON'T CARE

図7.43 出力タイミング

MSX-AUDIOの内部データ → MOの出力データ

Sign	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1		S_3	S_2	S_1	Sign	9	8	7	6	5	4	3	2	1
1	1	×	×	×	×	×	×	×	×							→	1	1	1	1	×	×	×	×	×	×	×	×	×
1	0	1	×	×	×	×	×	×	×	×						→	1	1	0	1	×	×	×	×	×	×	×	×	×
1	0	0	1	×	×	×	×	×	×	×	×					→	1	0	1	1	×	×	×	×	×	×	×	×	×
1	0	0	0	1	×	×	×	×	×	×	×	×				→	1	0	0	1	×	×	×	×	×	×	×	×	×
1	0	0	0	0	1	×	×	×	×	×	×	×	×			→	0	1	1	1	×	×	×	×	×	×	×	×	×
1	0	0	0	0	0	1	×	×	×	×	×	×	×	×		→	0	1	0	1	×	×	×	×	×	×	×	×	×
1	0	0	0	0	0	0	×	×	×	×	×	×	×	×	×	→	0	0	1	1	×	×	×	×	×	×	×	×	×
0	1	1	1	1	1	1	×	×	×	×	×	×	×	×	×	→	0	0	1	0	×	×	×	×	×	×	×	×	×
0	1	1	1	1	1	0	×	×	×	×	×	×	×	×		→	0	1	0	0	×	×	×	×	×	×	×	×	×
0	1	1	1	1	0	×	×	×	×	×	×	×	×			→	0	1	1	0	×	×	×	×	×	×	×	×	×
0	1	1	1	0	×	×	×	×	×	×	×	×				→	1	0	0	0	×	×	×	×	×	×	×	×	×
0	1	1	0	×	×	×	×	×	×	×	×					→	1	0	1	0	×	×	×	×	×	×	×	×	×
0	1	0	×	×	×	×	×	×	×	×						→	1	1	0	0	×	×	×	×	×	×	×	×	×
0	0	×	×	×	×	×	×	×	×							→	1	1	1	0	×	×	×	×	×	×	×	×	×

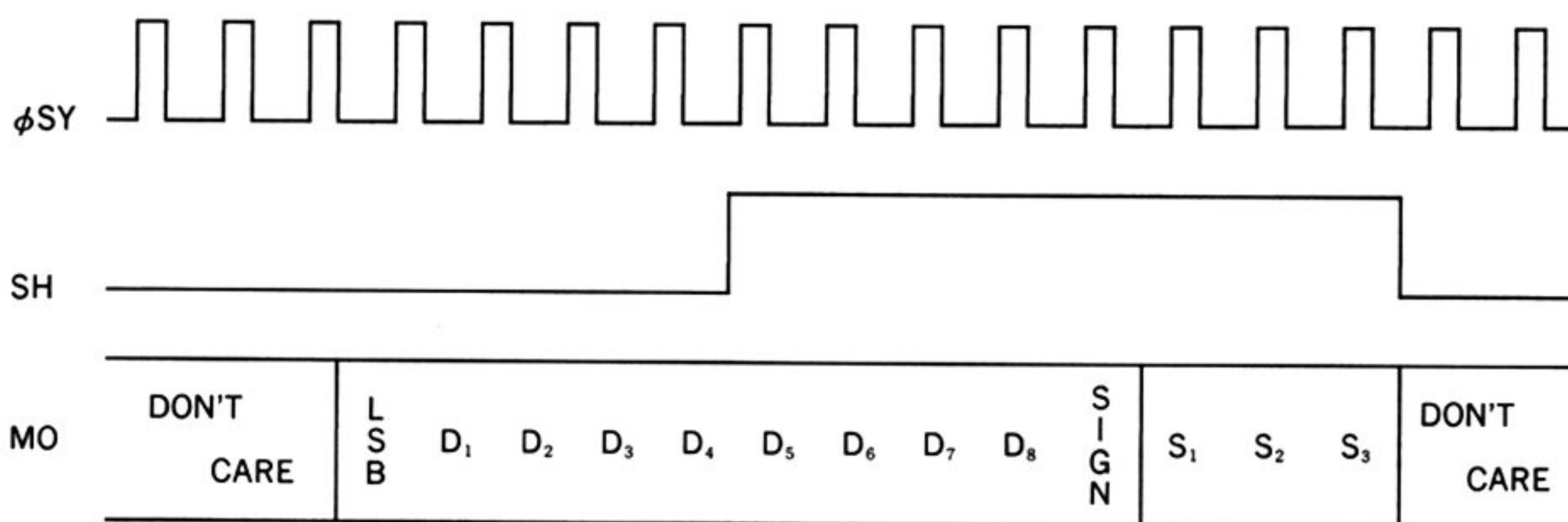
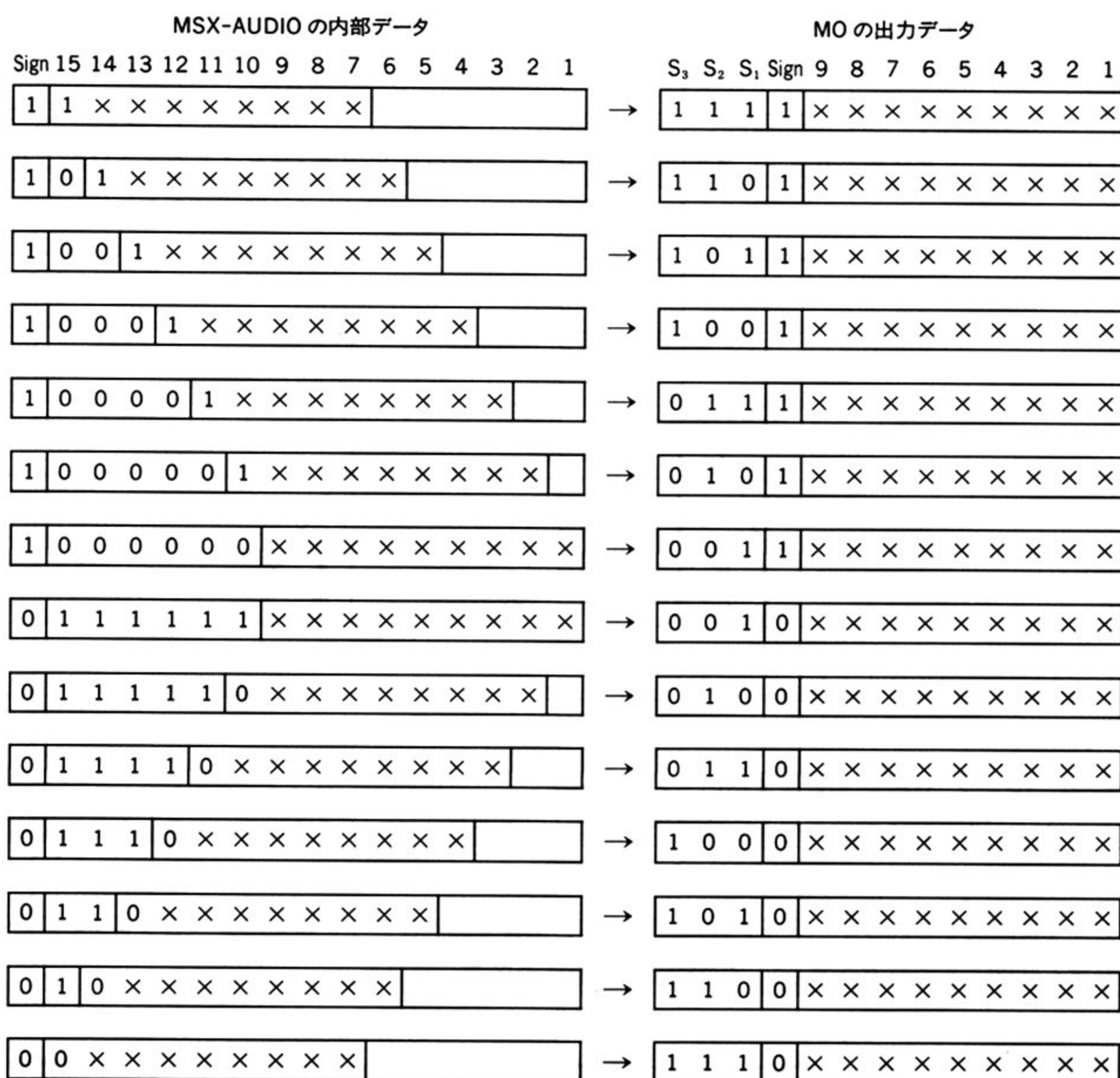



図7.44 内部データと出力データ

5. ADPCM 音声分析・合成

ここでは、音声分析・合成の方法をレジスタとのやりとりを例にしたがって説明します。

1. 音声分析（AUDIO → CPU）

アドレス	データ	R/W	コメント
			○初期設定
$04	$00	W	各フラグをイネーブルにする。
$04	$80	W	各フラグをリセット。
$07	$C8	W	ADPCM分析をイネーブルとし、スピーカをOFFとする。
$08	$00	W	
$0D	$C2	W	サンプリングレートを 8KHz（N_{PRE}= 450）。
$0E	$01	W	
			○分析スタート
$0F		R	ダミーリードにより分析開始
			○分析中
$0F ($04	 $80	R W)	BUF･RDY フラグが「1」のとき、$0F をリードして分析データを格納し、フラグリセット。「0」であれば待機。
			○分析終了
$07	$48	W	ADCPM 分析終了。
$07	$00	W	$07 レジスタリセット。

2. 音声合成（CPU → AUDIO）

アドレス	データ	R/W	コメント
			○初期設定
$04	$00	W	各フラグをイネーブルにする。
$04	$80	W	各フラグをリセット。
$07	$80	W	ADPCM 合成をイネーブルにする。
$08	$00	W	
$10	$F6	W	サンプリングレートを 8KHz（ΔN = 10486）。
$11	$28	W	
$12	$□□	W	出力レベル設定
			○合成スタート
$0F	$××	W	ADPCM データを$0F に書き込むことによりスタート。
			○合成中
$0F ($04	$△△ $80	W W)	BUF･RDY フラグが「1」のとき、$0F に合成データを書き込み、フラグをリセットする。「0」のときは待機。
			○合成終了
$07	$00	W	ADPCM 合成終了。

3. 音声分析（AUDIO → EXT.MEMORY）

アドレス	データ	R/W	コメント
			○初期設定
$04	$08	W	各フラグをイネーブルにする。
$04	$80	W	各フラグをリセット。
$07	$68	W	ADPCM分析をイネーブルにする。
$08	$02/$00	W	RAMタイプの指定。
$09	$××	W	メモリのスタート番地
$0A	$××	W	
$0B	$△△	W	メモリのストップ番地
$0C	$△△	W	
$0D	$E1	W	サンプリングレートを16KHz（N_{PRE}= 255)。
$0E	$00	W	
			○分析スタート
$07	$E8	W	$07のD7が「1」になるのに同期して分析開始。
			○分析中
			EOSフラグが「1」となり、分析終了を指示するまで待機。
			○合成終了
$07	$68	W	ADPCM分析終了。
$07	$00	W	$07レジスタリセット

4. 音声合成（EXT.MEMORY → AUDIO）

アドレス	データ	R/W	コメント
			○初期設定
$04	$08	W	BUF・RDYフラグのみマスクする。
$04	$80	W	各フラグをリセット。
$07	$20/$30	W	ADPCM分析をイネーブルにする。
$08	$00、$01、$02	W	RAMタイプの指定。
$09	$××	W	メモリのスタート番地
$0A	$××	W	
$0B	$△△	W	メモリのストップ番地
$0C	$△△	W	
$10	$EC	W	サンプリングレートを16KHz（ΔN = 20972)。
$11	$51	W	
$12	$□□	W	出力レベルの設定
			○合成スタート
$07	$A0/$B0	W	$07のD7が「1」になるのに同期して合成開始。
			○分析中
			EOSフラグが「1」となり、分析終了を指示するまで待機。
($07	$00	W	リピートモードを解除
$07)	$00	W	合成を強制的に中止)

アドレス	データ	R/W	コメント
			○合成終了
$07	$20	W	ADPCM 合成終了
$07	$00	W	$07 レジスタ終了

6. AD/DA 変換

内蔵の AD/DA 変換器は、FM 音源や ADPCM 音声分析・合成以外にも単独で AD 変換器、あるいは DA 変換器として使うことができます。この場合の変換速度は最大サンプリングレート 16KHz から最小 1.8KHz までです。

1. AD 変換

MSX-AUDIO の AD 変換は、音源で使用されている DA 変換器を利用して行います。したがって、この DA 変換器の制限から変換可能な電圧範囲は Vcc/2±Vcc/4 となります。Vcc/2 が中点で 0、3Vcc/4 が最大 (127)、Vcc/4 が最小 (−128) の 2 の補数 8 ビットデータに変換します (変換方式は逐次比較変換)。

なお、AD 変換中は、ミュージック出力などの DA 変換器につながる機器は切り離していないと大音量を発生するなどの問題が生じます。

2. DA 変換

DA 変換も、AD 変換と同様に楽音用の DA 変換器を共用します。したがって、出力電圧は Vcc/2±Vcc/4 となります。DA 変換は指数部 3 ビット、仮数部 10 ビット、計 13 ビットのデータですが、AD 変換などとの対応で 8 ビットで処理したい場合は、$15 アドレスのデータのみを可変とし、$16、$17 のアドレスは一定値に固定すればバイト処理ができます。

7. 外部メモリコントロール

MSX-AUDIO では ADPCM 音声分析・合成部のデータファイルとして外部メモリを RAM256K バイト、ROM256K バイトまでアクセスできます。この外部メモリの制御、およびデータのインターフェイスを行うのが外部メモリコントロール部です。

1. RAM

RAM は 64K DRAM、256K DRAM のいずれかを 8 個まで外付けできます。この場合、アクセスは 1 番目の RAM から 8 番目の RAM まで順に、また 1 個の RAM 内で次の図のように、(0,0) 番地から (511,0)、(0,1)～(511,511) とシリアルに READ/WRITE します。したがって、RAM でのデータ処理はビット単位となり、アドレス指定は 32 ビット (4 バイト) 単位になります。

なお、RAM のリフレッシュは MSX-AUDIO にカウンタを内蔵しており、自動的にアドレス発生を行います。

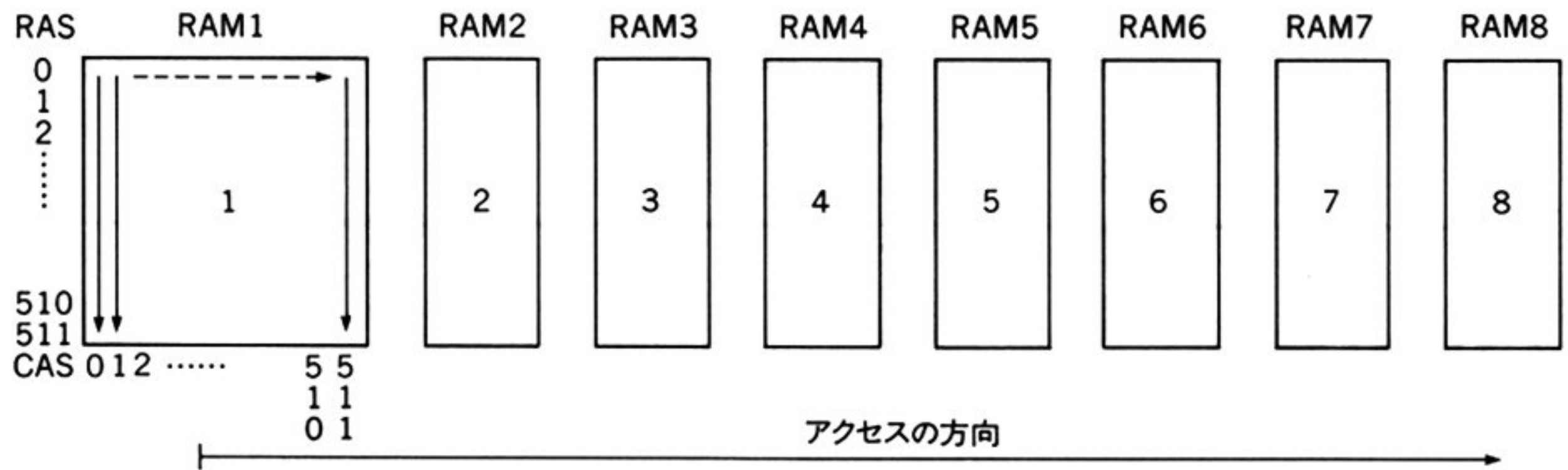



図7.45 RAMのアクセス

2. ROM

ROMの場合は、各アドレスにRAMと違ってMSX-AUDIOのDM出力を直接接続するのではなく、LATCHを介してROMにアドレスを入力します。また、アクセスの単位はバイト単位となり、アドレス指定は32バイト毎に設定できます。

3. メモリへのアクセス

メモリへのアクセスはADPCM実行中はMSX-AUDIOが自動的に行いますが、CPU側とデータのやり取りをする場合は、次の例にしたがってプログラミングします。

1. RAM —— WRITE

アドレス	データ	R/W	コメント
			○初期設定
$04	$00	W	各フラグをイネーブルにする。
$04	$80	W	各フラグをリセット。
$07	$60	W	メモリライトモードにする。
$08	$00/$02	W	メモリのタイプ指定
$09	$××	W	スタートアドレス指定
$0A	$××	W	
$0B	$△△	W	ストップアドレス指定
$0C	$△△		
			○メモリライト
$0F	$□□	W	データの書き込み。
($04	$80	W	BUF・RDYフラグが「1」のときデータ書き込み、「0」のときは待機。EOSフラグが「1」になると書き込み終了。)
			○リセット
$07	$00	W	$07レジスタリセット

2. RAM/ROM —— READ

アドレス	データ	R/W	コメント
			○初期設定
$04	$00	W	各フラグをイネーブルにする。
$04	$80	W	各フラグをリセット。
$07	$20	W	メモリリードモードにする。
$08	$00,$01,$02	W	メモリのタイプ指定
$09	$××	W	スタートアドレス指定
$0A	$××	W	
$0B	$△△	W	ストップアドレス指定
$0C	$△△		
			○メモリリード
$0F		R	ダミーリードを2回するとメモリのデータの読み出しを開
$0F		R	始する（フラグを見る必要がある)。
$0F	$□□	R	データの読み込み。
($04	$80	W	BUF•RDY フラグが「1」のときデータ読み込み、「0」のときは待機。EOS フラグが「1」になると書き込み終了。)
			○リセット
$07	$00	W	$07 レジスタリセット

8. KEY BOARD IN/OUT

キーボード入力・出力はダイオードマトリックスによる鍵盤を接続するのに便利なように、入力側にプルアップ抵抗を持ち、出力側はオープンドレインとなっています。入出力は各8ビットあるため、49鍵のキーボードまで接続が可能です。また、ドライブ能力は20μsのスキャニングレートで500pFの負荷まで適用でき、汎用の入出力ポートとしても利用できます。

9. ステータス情報とインタラプト信号

MSX-AUDIOのステータス情報は、2つのタイマからのフラグとADPCM音声分析合成や外部メモリアクセス時に使われる2つのフラグ（BUF•RDY、EOS）があります。これらのフラグは、そのイベントが起こったときに「1」になります。また、不必要なフラグに対しては、マスクすることもできます。

これらのステータス情報は、インタラプト信号につながっており、いずれかのフラグが「1」となったとき、インタラプト信号(IRQ)はLOWレベルになります。このインタラプト信号はオープンドレイン出力ですから、他の機器のそれとワイアードオアをとることができます。

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
IRQ	TIMER-1	TIMRE-2	EOS	BUF・RDY			PCM・BSY

D_0 (PCM・BSY)

ADPCM 音声分析・合成が実行中を表します。アドレス$07 の D7 (START) が「1」になると、このビットも「1」になります。この信号はインタラプトは発生しません。

D_3 (BUF・RDY)

このビットは次のとき「1」になります。

- ADPCM 音声分析　　2データ分析終了時 (@ADD.$07 の D_5= 0)
- ADPCM 音声合成　　2データ合成終了時 (@ADD.$07 の D_5= 0)
- 外部メモリライト　　1データメモリライト終了時
- 外部メモリリード　　1データメモリリード終了時

D_4 (EOS)　ADPCM 音声分析・合成実行中に、その分析・合成が終了したとき、あるいは AD/DA 変換時にそのサンプリング時間が経過したときに「1」となります。

D_5 (TIMER-2)

タイマ2によるフラグです。タイマ2のセットされた時間が経過したときに「1」になります。

D_6 (TIMER-1)

タイマ1に対して D5 と同様の働きをします。

D_7 (IRQ)　D3～D6 のいずれかが「1」のとき、「1」になります。

4.5.4 楽音の作り方

この章では、MSX-AUDIOのオリジナル音色レジスタに、どのようなDATAを入力すると、ピアノやブラスなどの楽音を作ることができるかを説明します。

1. 音作りの考え方

FM方式での音作りの基本は、まず作りたい楽器の特徴をよく理解することです。例えば、ピアノであれば、鍵盤を押したときに、鋭い音の立ち上がりがあり、その後、押鍵を続けていれば、徐々に音が消えて行くエンベロープを持っています。また、倍音の構成も立ち上がり時に多く、時が経つに連れて倍音の数は少なくなり、一定の倍音構成に近づいて行きます。

以上のような特徴をつかんだ後、FMの式で以下にして実現するかを考えます。エンベロープの特徴から出力振幅を、そして倍音構成から変調指数を決めることができます。また、倍音の構成はオペレータの周波数も関与していますから、周波数比もある程度決めることができます。このように、各楽音の特徴からFMの各パラメータをおおまかに決め、その次に音を聞きながら細部をつめてゆくようにすれば、望みどおりの音色を得ることができます。

2. 音作りの基本

FM音源とは、モジュレータによってキャリアを変調することから生じる効果を利用したものです。したがって、FMの基本式パラメータ(キャリアの出力レベル、モジュレータの出力レベルモジュレータのフィードバックレベル、キャリアの周波数、モジュレータの周波数) を上手に扱うことにより、各楽音のピッチ、音色、音量のすべてを決めることができます。このFMの各パラメータとMSX-AUDIOのパラメータとの関係は、表7.95のとおりです。

1. FM接続 (CONNECTION = 0)

表7.95のFMの特徴がすべて表現できます。また、オペレータ1はそれ自体にフィードバックがかかっているので、オペレータ2との組み合わせによる高周波の出方は2段のFM接続としての効果が得られます。

2. パラレル接続 (CONNECTION = 1)

2つのオペレータの足し算となり、オペレータ2は常にSIN波を発生させます。したがって、2つのオペレータの周波数をハーモニックにずらすことによりパイプオルガンのカプラー効果などを表現することができます。また、オペレータ1はFM接続と同様、フィードバックを持っているので高調波を出すことができます。

3. 音作りの例

表 7.95 音作りの基本

項 目	関与するパラメータ	MIN ←(音の変化)→ MAX
キャリアの出力レベル	TOTAL LEVEL (A・D・S・Rの各データ Key Scale データ)	音量小 ←→ 音量大
モジュレータの出力レベル		丸い音色 ←→ 明るい音色
モジュレータのフィードバックレベル	FB	普通の音色 ←→ 鋭い音色 (Noise)
キャリアの周波数	MULTIPLE	ピッチ低 ←→ ピッチ高
モジュレータの周波数	(BLOCK / F-Number)	近い倍音 ←→ 離れた倍音

1. エレクトリックピアノ

a. コネクションの選択

コネクションは「0」を設定します。ほとんどの音色はこのコネクションで得られます。ここでは、オペレータ1がアタック時のアクセントとリッチな高調波の両方を創り出します。

b. オペレータの周波数の決定

整数倍の高調波をすべて出すために、2つのオペレータともにMULTIPLEは「1」を使います。

c. オペレータの出力レベル

今度はモジュレータの出力を変更して音色を調整します。このとき、オペレータ1のレベルを決めるときには、低音部がまずピアノらしいリッチな高調波を得られるように設定し、それから高音にかけての変化はオペレータ1のレベルスケーリングで調整します。高音部ではほとんどSIN波になる位までレベルスケーリングをする必要があります。

d. EGの設定

ここでは音量と音色のエンベロープを決めます。まず、オペレータ2はアタックを鋭く、しかもある程度長く伸びるエンベロープにします。モジュレータになるオペレータ1では立ち上がりだけ倍音が多く、あとは一定にして音色変化はさせません。音量調整としてオペレータ2についてもキースケーリングをかけます。また、高音部にかけて音のシャープさを出すためには、RATEのスケーリングを行うとよいでしょう。

e. データの再調整

以上で音作りはほぼ終了ですが、EGなどのセッティングにより音色が幾分違ったものになってきます。この場合、オペレータの出力レベルやフィードバックレベルを再調整して、最終的な

音に仕立てます。例えば、金属的な響きが強すぎると思われる場合には、オペレータ1のレベルを下げます。

f. エフェクト付け

最後にエレクトリックピアノの音をより生かすために、トレモロ効果をLFOによってつけ加えます。これは内蔵の振幅変調の機能を利用してもよいですし、ソフトウェアでTOTAL LEVELの値を2〜6Hzの周期で更新（三角波で可）することも可能です。

2. トランペット

a. コネクションの選択

ブラス系のコネクションも「0」です。オペレータ1のフィードバックレベルをコントロールすることにより、ブラス系の派手な音作りが可能です。

b. オペレータ出力

モジュレータであるオペレータ1のトータルレベルは$10〜$28程度の控えめな値にし、フィードバックレベルはブライトな響きを出すために最大の「7」にします。

c. オペレータの周波数

基本的には、両方のオペレータ共に1倍にセットすればよいでしょう。

d. EG

2つのオペレータとも、ゆっくりとしたアタック音にします。そしてブラスのサウンドではモジュレータのアタックはすべてキャリアよりも遅くします。「ファン」というブラス特有のアタックを表現するのに必要なことです。

e. キースケーリング

ゆっくりとした立ち上がりにエンベロープをセットしたため、高音部でハギレが悪くなります。このため、速いパッセージを弾いたときに不自然にならないように、レイトスケーリングを少しかけます。

f. LFO

ブラスはどんな上手なプレーヤーが吹いても、ロングトーンの場合にはピッチがほんの少し揺れてきます。これを表現するためにビブラート効果を加えます。

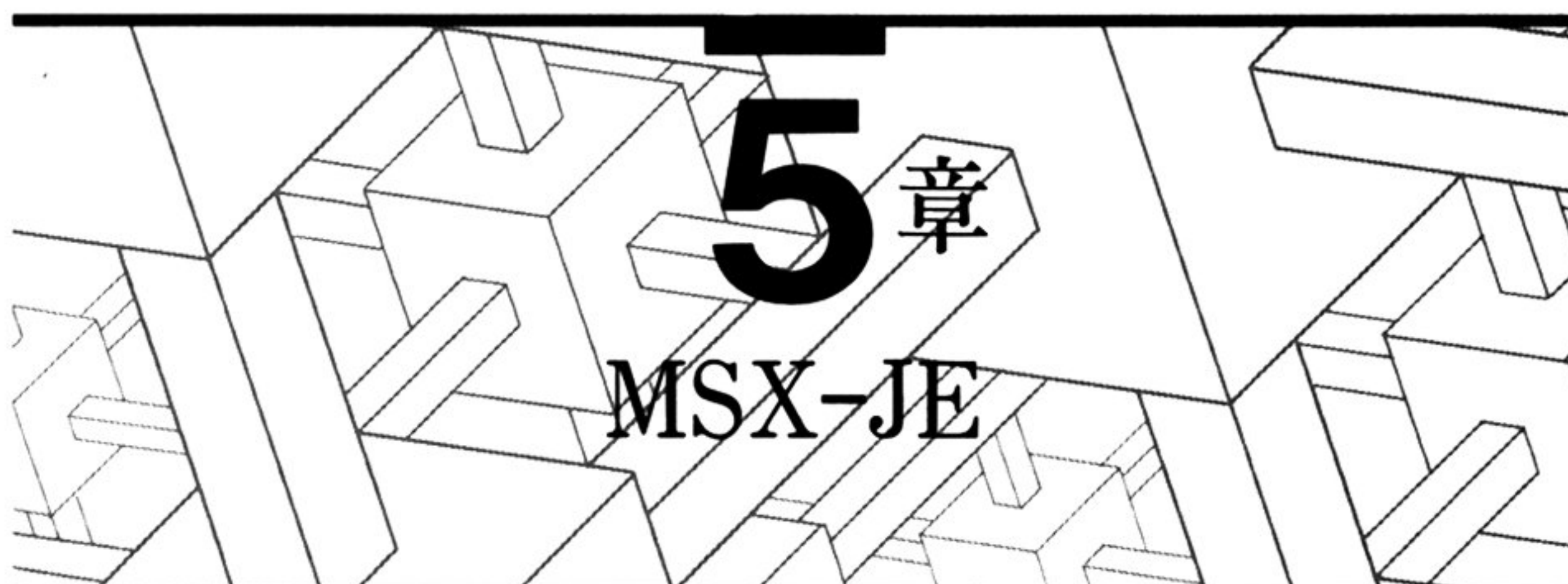
3. リズム音の作り方

リズム音は7、8、9のチャンネルを使って作られます。この3チャンネル6スロットで計5音のリズム音を作るわけですが、バスドラム(BD)のみは2スロットでFM音を作ります。したがって、バスドラムについては(a)〜(c)で述べたことと基本的には同一手法で作ることができます。

そこで、ここでは残り4音（ハイハット、トップシンバル、タム、スネアドラム）について説明します。

MSX-AUDIOには、リズム楽器のためにホワイトノイズジェネレータと数種の周波数を合成して得られるノイズ発振器があります。このノイズ発振器は8チャンネルと9チャンネルの周波数情報（BLOCK、F-Number、MULTIPLE）より作られ、ホワイトノイズと合成することにより各リズム楽器に適した位相出力を発生して、オペレータに渡します。つまり、ここでは2つの周波数情報から4つの楽器の位相を作っていることになります。

なお、2つの設定周波数は経験的に3：1（f7CH＝3×f8CH）が良いとされています。これで、各楽器の位相データが得られたことより、この出力にエンベロープの情報を掛け合わせます。エンベロープは1スロットに1リズム楽器と設定されているため、メロディ楽器同様各リズム楽器の特徴をつかんだ値を各パラメータレジスタにセットします。「AM・VIB-DEPTH/RHYTHM」を参照して下さい。

5章 MSX-JE

5.1 概要

MSX-JE 仕様とは、MSX 日本語入力フロントエンドプロセッサのインターフェイスを規定するものです。これは、MSX 上のアプリケーションソフトウェア（以下、AP）と日本語入力処理を分離し、容易に日本語入力を行えるようにするためのインターフェイスです。

MSX-JE の仕様に沿って作成することにより、図 7.46 のように AP といろいろな種類のフロントエンドプロセッサ（以下、FEP）を自由に組合せることができます。そして、複数の AP で同一の FEP を共有したり、AP を変更しないで、FEP のみをより高機能なものに発展させたりすることが可能になります。

<table>
<tr><td colspan="3">日本語
ワード
プロセッサ</td><td colspan="3">日本語
統合
ソフトウェア</td><td colspan="3">日本語
表計算
ソフトウェア</td><td colspan="3">日本語
通信
ソフトウェア</td></tr>
<tr><td colspan="12">MSX 日本語入力フロントエンドプロセッサ
インターフェイス</td></tr>
<tr><td colspan="4">単語変換型
フロントエンド
プロセッサ</td><td colspan="4">ワープロ普及機型
フロントエンド
プロセッサ</td><td colspan="4">一括変換型
フロントエンド
プロセッサ</td></tr>
</table>

図 7.46　MSX-JE の概念図

MSX-JE は大きく分けて、

1. 仮想端末入力インターフェイス
2. 辞書インターフェイス

の2つから構成されています（図7.47参照）。

仮想端末入力インターフェイスを使用するときは、かな漢字変換入力時にはAPは単にスクリーンへのメッセージ出力を行うだけです。つまり、APは、BIOSの文字入力を使用するのと同じような手軽さで、日本語入力を行うことができます。したがって、APは日本語入力の複雑な処理をFEP側にまかせて、本来の目的に専念することができます。ワープロ以外の、日本語入力を主体としないAPには、この仮想端末入力型インターフェイスが適しています。

また、日本語ワープロなどで仮想端末入力インターフェイスに頼らず、独自のユーザーインターフェイスを持ちたい場合は、辞書インターフェイスを使用することができます。辞書インターフェイスはこの他にも、例えば、ノンインタラクティブにかな漢字変換を行うプログラムなどにも利用できます。

簡単なAP（メモパッドやグラフツールなど）
高度なAP（日本語ワープロなど）
画面出力
仮想端末入力インターフェイス
ユーザーインターフェイス
キーボード入力
辞書インターフェイス
辞書アクセス
MSX日本語入力フロントエンドプロセッサ（MSX-JE）

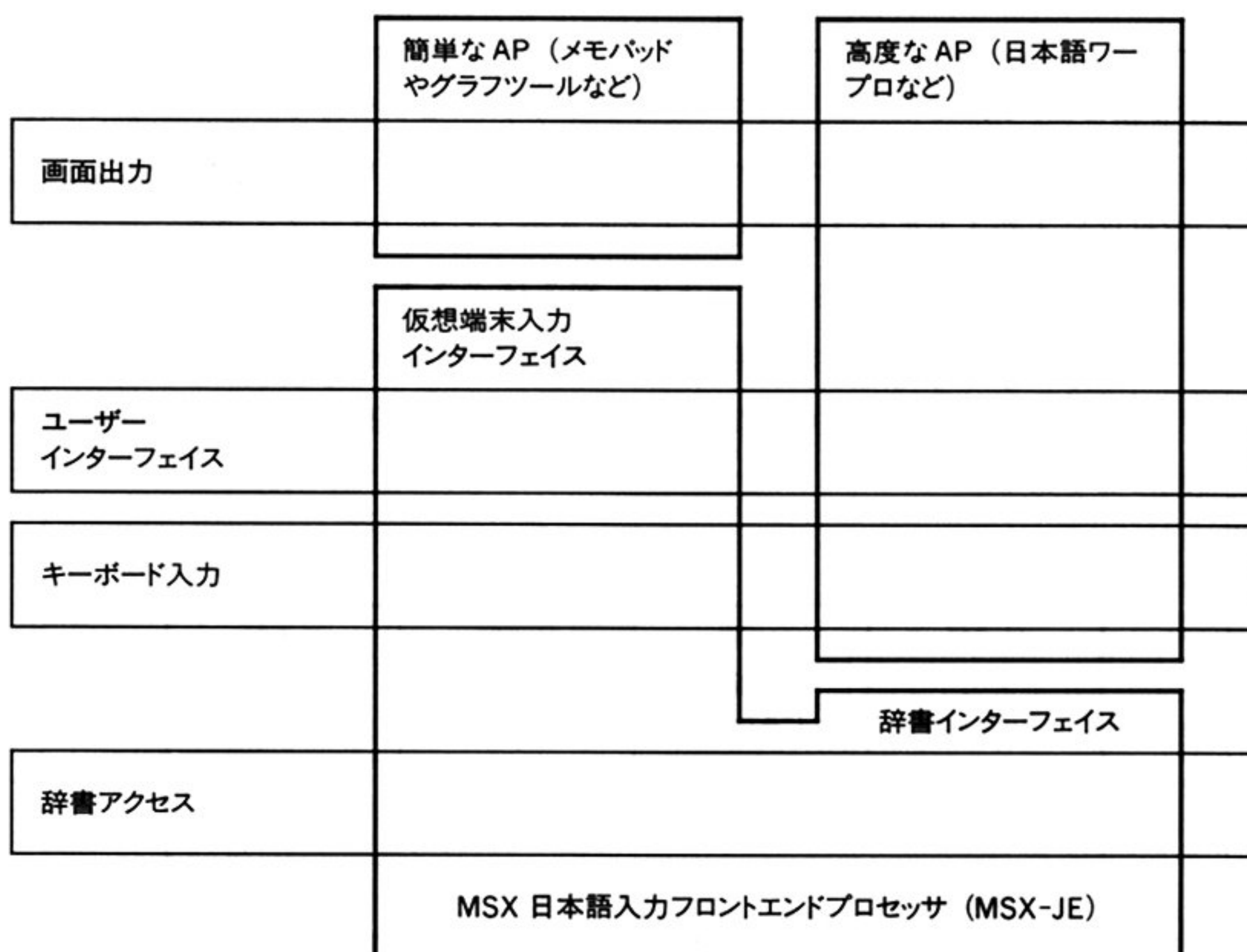



図7.47　MSX-JEの構成

5.2 MSX-JEの使用規定

5.2.1 適用範囲

この仕様の適用対象は、表計算ソフト、データベース、ゲームなど、外部のMSX-JEを利用して日本語入力を行うアプリケーションソフトウェアです。これらのアプリケーションソフトウェアはこの仕様を満たしていれば、自分でかな漢字変換を行わなくても、MSX-JEを使用して日本語入力を行うことができます。

MSX-JEは、

1. 日本語カートリッジ（ソニー）などの独立したカートリッジ
2. HB-F1XV（ソニー）、FS-A1WSX（パナソニック）などのようにMSX本体に内蔵
3. MSX-WriteII（アスキー）など、日本語ワープロがかな漢字変換機能だけを外部に提供

などの形式で持つことができます。ワープロソフトがこのようなインターフェイスを持つことにより、ワープロを購入したユーザーは他のアプリケーションソフトウェアでも、ワープロの日本語入力機能を利用して、かな漢字変換入力ができることになります。

このような利用方法を可能とするために、以上にあげたようなソフトウェアは、MSX-JEの仕様にそったインターフェイスを持たせることを推奨します。

5.2.2 日本語ワープロの場合

日本語ワープロのようにワープロと辞書が1つのROMカートリッジに一体化されている場合、そのワープロと辞書とのインターフェイスには、このインターフェイスを遵守する必要はありません。この場合、このインターフェイスは外部につけられた他のアプリケーションとの間でのみ使用されることになります。

内部に辞書をもつ日本語ワープロなどは、このインターフェイスを用いて外部からその辞書をアクセスできることを推奨します。

5.3 ファンクションコールの方法

MSX-JE は大きく分けて、以下の 2 レベルの機能をサポートします。

1. 仮想端末入力インターフェイス

仮想端末上でかな漢字変換を行う。

2. 辞書インターフェイス

アプリケーションが直接辞書をアクセスするためのインターフェイス。

これらはファンクション番号によって区別されます。

5.3.1 エントリアドレスの獲得

AP はまず最初に MSX-JE が実装されていることを確認し、MSX-JE のエントリアドレスを調べなければなりません。これは、MSX の拡張 BIOS コールで以下のように行います。

1. AP はまず【HOKVLD(FB20H)】のビット 0 を調べます。ここが 0 の場合、拡張 BIOS は存在せず、したがって MSX-JE もありません。この場合、EXTBIO をコールすることはできません。
2. 次に、RETURN 情報用のワークエリア（64 バイト）を確保します。
3. 以下の設定を行い、【EXTBIO(FFCAH)】をコールします。

コール手順

D	デバイス番号（16） MSX-JE のデバイス番号は 16（10 進）
E	ファンクション番号（0）
B	RETURN 情報用のワークエリアのスロットアドレス 上記の 2.で確保したワークエリアの RAM のスロットアドレス
HL	RETURN 情報エリアの先頭アドレス

RAM のスロットアドレスは以下のワークエリアに保存されています。このワークエリアはディスクが接続されているシステムで有効です。ディスクが接続されていないシステムで RAM のスロットアドレスを捜す場合は、サンプルプログラムの「RAMSRCH.MAC」を参照して下さい。

ページ	ワークエリアのアドレス
0	F341H
1	F342H
2	F343H
3	F344H

戻り値

B　　次の RETURN 情報エリアのスロットアドレス
HL　　次の RETURN 情報エリアの先頭アドレス

変更レジスタ

F

注　意

スタックはページ3になければなりません。

したがって、AP は以下のようにしてエントリアドレスを調べます。

リスト7.1　MSX-JE 拡張 BIOS エントリアドレスの獲得

```
hokvld  equ     0fb20h          ; Extended BIOS flag
extbio  equ     0ffcah          ; EXtended BIOS hook
;
        ld      a, (hokvld)
        rra
        jr      nc, no_mje
        ld      hl, mjetbl
        call    getslt          ; Get slot address of MJETBL into [B]
        ld      de, 16*256+0
        push    hl
        call    extbio
        pop     de
        or      a
        sbc     hl, de
        jr      z, no_mje
        .
        .
        .
```

EXTBIOから戻ったとき、MJETBLには以下のようなデータが返され、HLはMSX-JEのエントリ分先に進められます。MSX-JEが存在しないときは、HLの内容は変更されません。この例はAとBの2つのMSX-JEが存在している場合です。

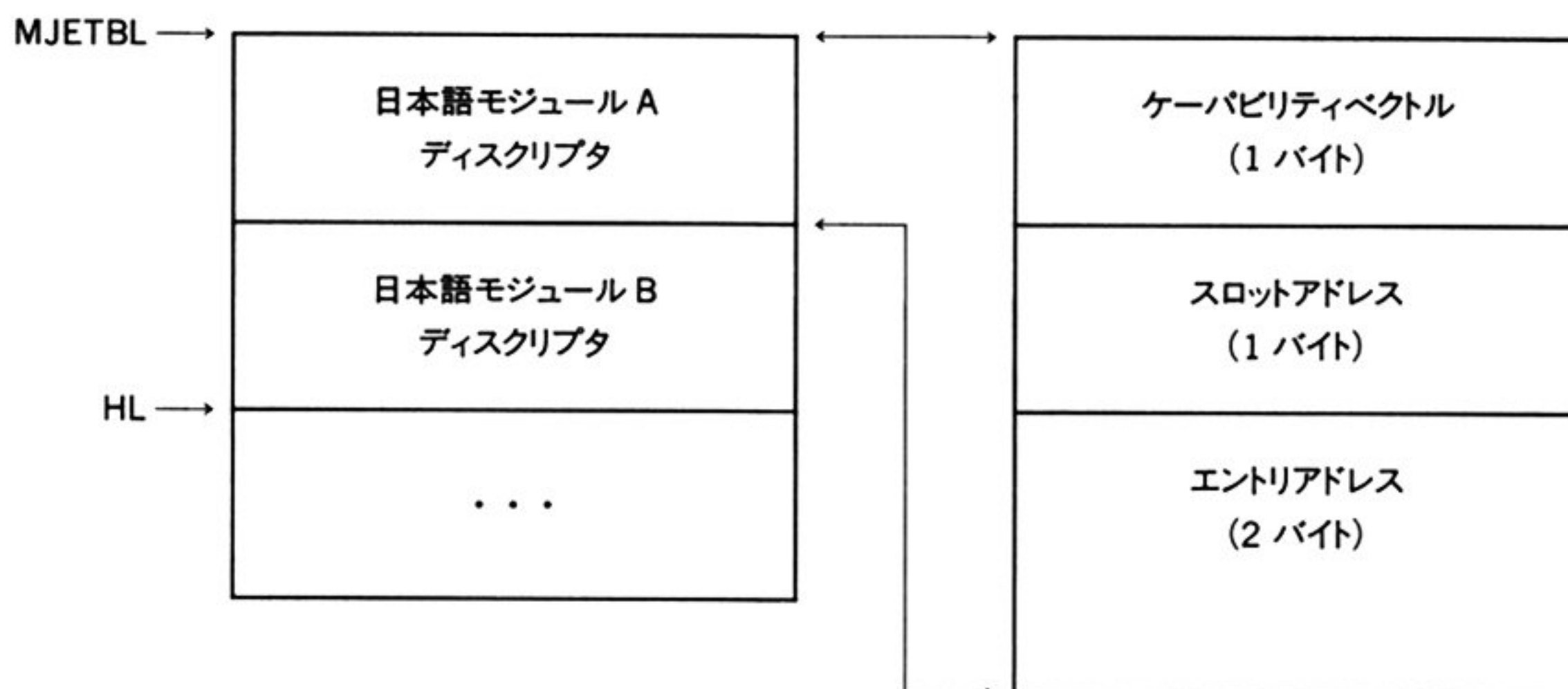



図7.48 RETURN情報の形式

スロットアドレスの表現はMSX共通で以下の通りです。

7	6	5	4	3	2	1	0
F	0	0	0	S	S	P	P

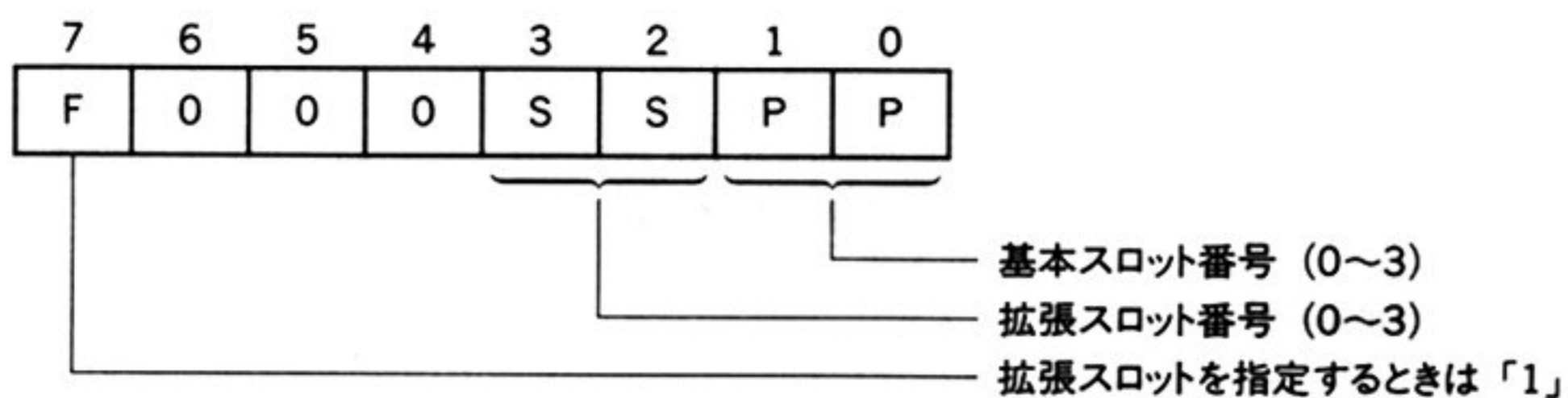



図7.49 スロットアドレスの形式

拡張BIOSを使用する場合は、この拡張BIOSコールで得られたジャンプテーブルをインタースロットコールなどにより呼び出し、目的のBIOSを使用します。

ケーパビリティベクトルは以下のようなビットパターンです。

7	6	5	4	3	2	1	0
1	1	1	1	X	0	0	0

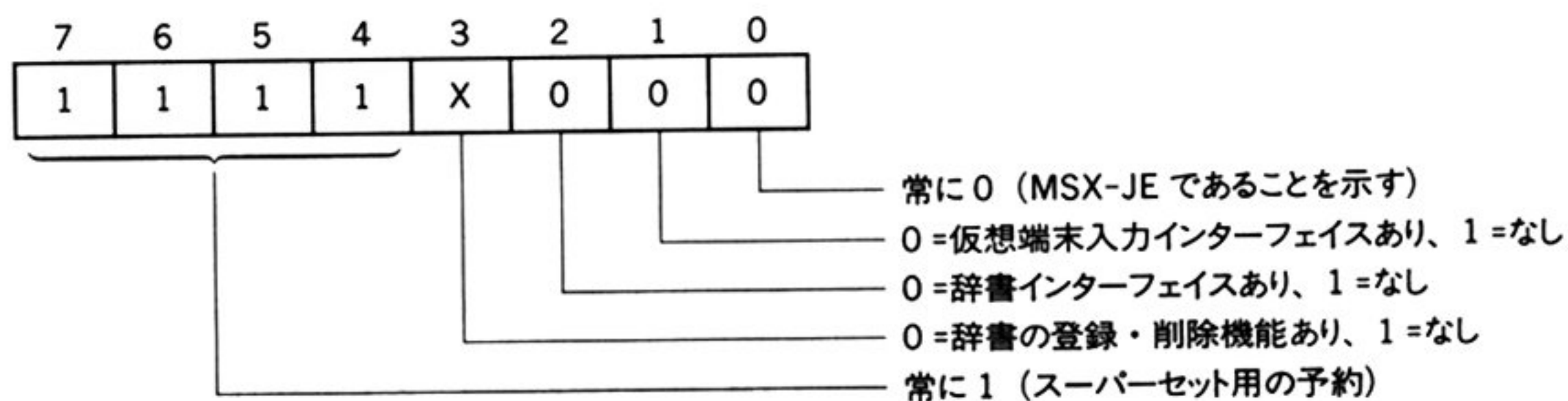



図7.50　MSX-JE のケーパビリティベクトル

このビット列は、装備されている日本語処理機能のサブセットとスーパーセットを示すためのビット列です。

AP はこのビット列をチェックし、利用可能な機能を確認した上で MSX-JE をコールして下さい。例えば、仮想端末インターフェイスのみを使用する AP はこのビット列と 03H との AND を取り、ビット 0 と 1 が 0 になっていることを確認します。

MSX-JE 側では、MSX のイニシャライズの時点で、自分の拡張 BIOS エントリを【EXTBIO (0FFCAH)】にフックします。拡張 BIOS では、DE レジスタが 1000H の場合に上記の動作をし、0 の場合にはブロードキャストの動作を行います。それ以外の場合は、表レジスタの内容を保存したままリターンします。ブロードキャストについては、「7章 拡張 BIOS コール」をご覧下さい。

5.3.2 MSX-JE の呼び出し

MSX-JE 側では固定番地にワークエリアを確保することが不可能なため、アプリケーションからコールされるごとに、自分のワークエリアの番地をレジスタにセットして渡してもらう必要があります。

MSX-JE をコールするときは以下のようにします。

コール手順

A	ファンクション番号
BC	パラメータ
DE	パラメータ
HL	ワークエリアの先頭アドレス

戻り値

A	(もしあれば) リターン値
BC	(もしあれば) リターン値
DE	(もしあれば) リターン値
HL	(もしあれば) リターン値

ワークエリア、および間接参照されるパラメータ(たとえばDEがストリングへのポインタだった場合、そのストリング)はページ2か3に置き、APはこれらのページを必ずイネーブルしてからMSX-JEをコールします。これは、ワークエリアがMSX-JEと同じページの裏側のスロットにおく事を許すとスロット切替が必要になり効率が期待できないためです。

MSX-JEではリターン値およびスタックポインタ以外のレジスタの値は保証されません。未定義のファンクションをコールした場合、なんの処理も行なわずにリターンします。ただし、スタックポインタ以外のレジスタの値は保証されません。

MSX-JEをコールする場合、HLの設定は必ず必要となるため、APはイニシャライズ時に次のようなプログラムを作成し、ファンクションコールにはこのプログラムを使用すると便利です。

```
mjea@:
mje@:   ld      hl, wrkbeg      ; ワークエリアアドレス
        rst     30h
        defb    sltadr          ; MSX-JE スロットアドレス
        defw    entadr          ; MSX-JE エントリアドレス
        ret
```

このプログラムはMSX-C言語から次のようにして呼び出せます。

1. HLにリターン値がセットされるファンクションの場合

```
int mje();
...
int bc, de, hl;
...
hl = mje(func, de, bc);
...
```

2. Aレジスタにリターン値がセットされるファンクションの場合

```
char mjea();
...
char a;
int bc, de;
...
a = mjea(func, de, bc);
...
```

5.4 仮想端末入力インターフェイス

以下ではMSX-JEの仮想端末入力インターフェイスの仕様を解説します。このインターフェイスの目的はAP側が日本語入力をサポートする場合の負担を軽くすることにあります。

かな漢字変換に関係するユーザーインターフェイスはMSX-JE側で受け持ちます。したがって、このインターフェイスを使用するとき、かな漢字変換入力時にはAPは単にスクリーンへメッセージを出力するだけです。つまり、APは、BIOSの文字入力ルーチンを使用するのと同じような手軽さで、日本語入力ができるようになります（図7.51参照）。

日本語ワードプロセッサのように、かな漢字変換時にも自分自身で独自のユーザーインターフェイスを持ちたい場合や、画面上でなくプログラムでかな漢字変換を行いたい場合は、辞書インターフェイスを使用します。

ユーザー
MSX-JE
AP
MSX-JE内部での変換作業
「aishite」
と入力
aishite
あいして
あいして
そのまま表示する。
あいして
変換キーを押す
愛して
そのまま表示する。
愛して
確定キーを押す
愛して
「愛して」が入力される。

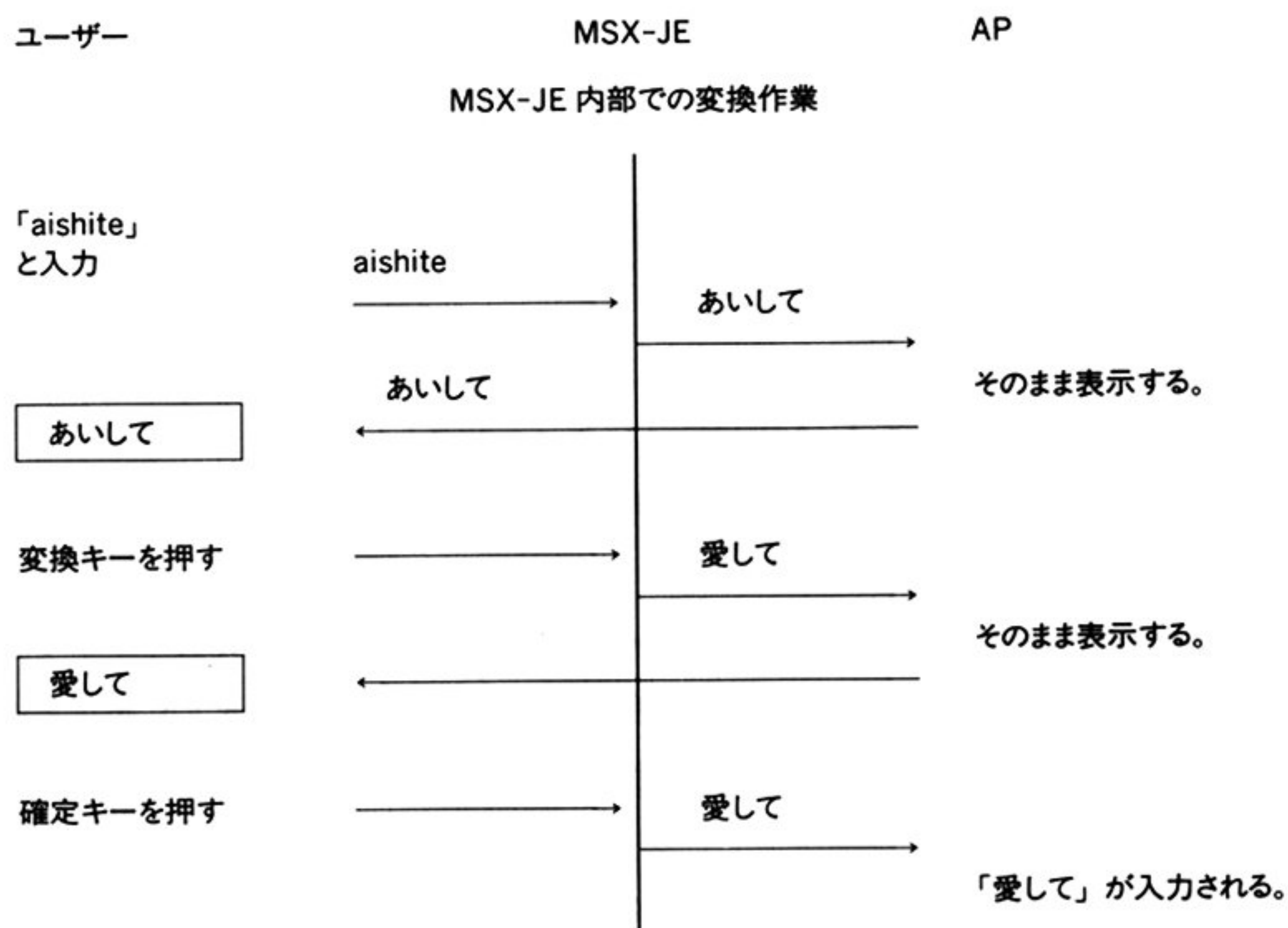



図7.51　MSX-JEの変換動作

5.4.1 キー割り当て

仮想端末入力インターフェイスでのキー入力処理は、かな漢字変換中はMSX-JEが行い、変換中でない場合はAPが行います。それぞれのプログラムはキーバッファを1文字先読みし、自分の受け持ちでない文字が入力されると処理を終了し、相手に制御を渡します。基本的には、コントロールコードはAPが、文字キーはMSX-JEが受け持ちます。

MSX-JEは機能を拡張するため、ファンクションキーの[F1]～[F5]を使用できます。ただし、制御の受け渡しはキーバッファのみを参照する方式になっているので、ファンクションキーを使うときは、あらかじめファンクションキーに特別なコードをセットしておかなければなりません。その場合、先頭1バイトに01H、2バイト目にキーの種別をセットします。01Hで始まる2バイトグラフィック文字と区別するため、以下のコードを使用します。なお、MSX-JEがファンクションキーを使わないのであれば、F1～F5をこのように設定する必要はありません。

[F1]	01H、31H
[F2]	01H、32H
[F3]	01H、33H
[F4]	01H、34H
[F5]	01H、35H

また、ファンクションキーの[F6]～[F10]はAPで使うことができます。その場合、先頭1バイトにはFFHをセットして下さい。APがファンクションキーを使用しない場合は、このように設定する必要はありません。また、特殊な使用方法として、よく使うキーストロークをAP側であらかじめ[F6]から[F10]に設定しておくなども可能です。

仮想端末インターフェイスでは、1文字先読みが可能なため、APとMSX-JEがそれぞれ相互に制御を渡すキーの範囲が異なると、どちらでも処理されないキーができてしまう可能性があります。そして最悪の場合、その様なキーが入力されたときに、処理が先に進まなくなってしまいます。

このような事態をさけるため、MSX-JEがバッファから取り出さないキーは必ずAPがバッファから取り出すよう、以下のようにキー割り当てを定めます。

■ AP

1. キーバッファから必ず取りだすキー

   コントロールコード（00H～1FH、7FH、ただし、01Hで始まる2バイトグラフィック文字およびF1～F5は除く）

   FFHで始まる、[F6]～[F10]に設定したキー

2. キーバッファから取り出さず、MSX-JEに制御を渡すキー
   上記以外のキー

■ MSX-JE

1. キーバッファから必ず取り出すキー
   すべての文字キー（スペースを含む）
   01Hで始まる2バイト
2. キーバッファから取り出さず、変換を終了させるキー
   FFH、[CTRL]+[C]

したがって、いったん変換が開始されるとMSX-JEはコントロールコードをコマンドとして使用することができます。しかし、この条件を守ると、コントロールコードで直接MSX-JEに制御を渡し、MSX-JEにコマンドを与えることはできません。

このような場合、Conflict detect（Function 10）をサポートすれば、APが自分で使用しない限り、コントロールコードをMSX-JEに直接入力できるようになります。

注　意

■ AP

MSX-JEのコールによって、ファンクションキー[F1]から[F5]の文字列の設定は変更されてしまいます。したがって、内容を保存する必要があればAPはClear（Function 4）またはSetTTB（Function 5）をコールする前に、自分でファンクションキーの内容をバックアップし、変換が終了したらファンクションキーを元の状態に戻して下さい。

APはキーキューバッファの先頭に01Hが入っていたときは、MSX-JEに制御を渡してかまいません。ただし、このままだと[CTRL]+[A]が処理できないので、これを避けるために、キーキューバッファに01Hのみがはいっていた場合、[CTRL]+[A]として取り扱うこともできます。

■ MSX-JE

ファンクションキーの文字列の設定はClearとSetTTBで行います。以前の設定をバックアップする必要はありません。

MSX-JEは、変換開始時にキーバッファに01Hのみが入っていた場合は、01Hをキーバッファから取り除き、APに制御を戻します。

5.4.2 APのプログラム例

リスト7.2 APのプログラム例

```
/* APのメインプログラム */

main()
{
        char work[2560] ;
        char c, MSX-JE_status ;

        /* MSX-JEのイニシャライズ */
        MSX-JE_Inquiry() ;
        MSX-JE_Invoke() ;

        /* かな漢字入力を始める */
        MSX-JE_Clear() ;

          for ( ; ; ) {
                /* キーバッファの先頭文字を先読みする */
                c = chsns() ;

                /* APが処理すべきキーかどうかのチェック */
                if (c == 0x01 | | 0x20 <= c && c != 0x7f && c != 0xff) {

                        /* 文字キーかF1～F5キーなら変換を開始する */
                        do {
                                MSX-JE_status = MSX-JE_Dispatch() ;
                            if (MSX-JE_status & 1) display(STB) ;
                            if (MSX-JE_status & 2)
                                    input(MSX-JE_GetResult()) ;
                        } while(!(MSX-JE_status & 4)) ;

                else {
                        /* APで処理する文字を取り出す */
                        c = chget() ;
                        /* 漢字入力の終わり */
                        if (c == xxx)
                                break ;
                        switch(c) {      /* 必要なら確定済文字列の編集 */
                        case ... :
                                ...
                        }
                }
        }
        /* MSX-JEを終了する */
        MSX-JE_Release() ;

}
/*  下位ルーチン  */
display(s)
char * s ;
{
/*  ポインタsの指しているSTB形式(後述)の表示データを
    かな漢字変換用ウィンドウに表示する。   */
```

```
}

input(s)
char ＊ s ;
{
/＊  ポインタｓの指している文字列をあたかもコンソールから
     入力されたように処理する。   ＊/
}
```

5.4.3 キーセンスの方法

APと仮想端末インターフェイスとの間でキーのやり取りをするためには、キーのセンスをして仮想端末に制御を移すべき文字なのかを調べる必要があります。しかし、BIOSのCHSNS(009CH/MAIN)では、キーバッファの状態を調べることはできても、その先頭の文字が何であるかを知ることはできません。そこで、以下の方法を取らなければなりません。

1. CHSNSでキーバッファの内容を調べる。
2. バッファが空でなければ、【GETPNT(0F3FAH, 2)】の内容が示すアドレスにあるデータが、次にCHGETしたときに得られるキーコード。

5.4.4 STB (Screen image Text Block)

STBはMSX-JEが画面表示を行うときに使用するデータ構造です。MSX-JEが日本語入力中に画面に出力する情報は、すべてSTBの形式でAPに渡されます。APはSTBのフォーマットを解釈して、あらかじめMSX-JE用に確保していたスクリーンの領域(ウインドウ)に文字列を出力します。

1. STBのフォーマット

表7.96 STBのフォーマット

データ	意　味
シフトJISコード、半角の英、数、仮名、記号文字	カーソル位置に文字を表示し、カーソルを文字幅分だけ進めます。
NUL (0)	テキストの終りです。
^E (5)	カーソル位置以降の文字を消去します。カーソル位置は変わりません。
^H (8)	カーソルを1文字後退し、カーソル位置以降の文字を消去します。
^L (12)	カーソルをウインドウの先頭に移動し、カーソル位置以降の文字を消去します。

データ	意　味
^R（18）	カーソル位置から次の1バイトで指定される長さだけ、ウインドウの領域をリバースします。
^W（23）	次の1バイトで指定されるウインドウを開きます。
^X（24）	次の1バイトをX座標としてカーソルを移動します。
^Y（25）	次の1バイトをY座標としてカーソルを移動します。
^Z（26）	アトリビュート指定です。次の1バイトの上位4ビットが文字色、下位4ビットが背景色です。
上記以外で00～15のコード	システム予約です。スキップして下さい。
上記以外で16～31のコード	システム予約です。このコードと続く1バイトもスキップして下さい。

注　意

■ **文字の位置指定**

文字の位置指定などはすべて半角単位です。また、X・Y座標の開始値は1です。

■ **カーソルの表示**

STBを表示するとき、カーソル移動を可能にした仮想端末のためにカーソル位置にカーソルを表示しなければなりません。このカーソルは、文字の指定と区別するために1を開始位置とし、X座標の指定が0のときはカーソルを表示しないものとします。

■ **ウインドウのクローズ**

STBには、ウインドウのクローズファンクションはないので、ウインドウのクローズはAPが管理しなければなりません。ウインドウのクローズはSTBがウインドウの消去（^L）のファンクションを返したときに可能です。

■ **ファンクションのサポート**

Inquire Window Size（Function 9）をコールしないAPは、STB表示フォーマットの^Yと^Zをサポートする必要はありません。

■ **以前のウインドウ**

STBはこの文字列を表示する前に、画面がどのような状態であったかはいっさい関知しません。したがって、ウインドウを開く前の状態にウインドウの画面を戻したいときには、ウインドウを開く前に、あらかじめ画面の内容をAPが保存しておかなければなりません。

■ **ウインドウ内の情報**

カーソル位置、文字のアトリビュート、ウインドウなどは、次回の表示まで保存され

ます。例えば、入力テキストの最後尾にカーソルがあり、そこに1文字追加したい場合、MSX-JE側では、追加文字だけのSTBを返せばよいことになります。

■入力テキストの表示

MSX-JE側では、複数のウインドウをサポートしないAPのため、入力テキストの表示は必ずウインドウ#0を使用しなければなりません。

関連項目

Dispatch (Function 6)

Inquire Window Size (Function 9)

5.4.5 TTB (Transferrable Text Block)

MSX-JEはAPと独立したプログラムとして動作します。したがって、MSX-JEのステータスの中で、変換途中の文字列の状態をAPがGetしたりSetできるように、TTBと呼ぶデータ構造を用意します。

TTBはAPがいったん変換したテキストの再変換を行うときに、読みデータを渡すために使用するデータ形式です。データは、Get TTB (Function 8) でMSX-JEからAPに渡され、Set TTB (Function 5) でMSX-JEにSetされます。

1. TTBのフォーマット

TTB::= TTC TTC ・・・, 0

TTC::= 1〜31　　1〜31バイトの読みデータ

32〜127　　1バイトの読みデータ

128〜255　継続読みデータ

TTBは「0」で終るTTCの列で、TTC1つが変換結果の1charに対応します。1つのTTCは、

- ■値が32〜127の1バイトの数
- ■値が128〜255までの1バイト数のくりかえしで、最後に値が32〜127の1バイトで終るもの
- ■値が1〜31の1バイトの後に、1〜31バイトの任意のデータが続いたもの

です。

例えば、ローマ字データを読みデータとするときはASCII文字列を使用し、変換結果1文字の最後に相当するローマ字以外はビット7を立てます。変換結果の漢字と読み番号で読みが得られるシステムなら、読み番号を32〜127の数字に変換してTTCにします。

TTCの各データの意味は特に規定しません。したがって、アプリケーションはTTCの表わしているものが、カタカナであるかローマ字であるかなどということは関知しません。TTCの意味的な解釈はMSX-JEだけが行います。

注 意

■ AP

Set TTB（Function 5）とGet TTB（Function 8）を使用しないAPは、TTBをサポートする必要はありません。

■ MSX-JE

MSX-JEがSet TTB、Get TTBファンクションをサポートしない場合は、TTBをサポートする必要はありません。

関連項目

Set TTB（Function 5）
Get TTB（Function 8）

5.4.6 ファンクション

表7.97 仮想端末インターフェイス一覧

ファンクション番号	ファンクション名	機 能
1	Inquiry	ワークエリアサイズの獲得
2	Invoke	MSX-JEの起動
3	Release	MSX-JEの終了
4	Clear	かな漢字変換バッファのクリア
5 [option]	Set TTB	再変換用データのTTBへの設定
6	Dispatch	MSX-JEに制御を渡す
7	Get Result	変換結果の獲得
8 [option]	Get TTB	テキストの読みデータの獲得
9 [option]	Inquire Window Size	ウインドウ形式の設定
10 [option]	Conflict detect	制御キーの衝突防止

注 意 [option]となっているファンクションは、APは使用しなくても良いが、使用することにより、より高度な処理が簡単にできるファンクションです。
また、ファンクション1～3は仮想端末インターフェイスと辞書インターフェイスの両方で使用できます。

Inquiry（Function 1）

機　能

ワークエリアサイズを獲得します。

コール手順

A　　01H

戻り値

HL　　MAX（MSX-JEが使用するワークエリアの大きさの上限）
DE　　MIN（MSX-JEが使用するワークエリアの大きさの下限）
BC　　MIN2（MSX-JEが学習機能を使用するのに必要最小なサイズ）

解　説

MSX-JEが動作するのに必要なワークエリアの大きさをAPに知らせます。MSX-JEは2560バイト以下のワークエリアで動作しなければなりません。したがって、MINに2560以上の数が返ることはありません。

注　意

■ AP

APがMSX-JEのために準備できるワークエリアの大きさをnとすると、必要なワークエリアの大きさは以下のようになります。

戻り値の関係					ワークエリアの大きさ
n	<	MIN			(APはMSX-JEを使用できない)
MIN	≦	n	<	MIN2	MIN（APは学習機能を使用できない）
MIN2	≦	n	<	MAX	n
MAX	≦	n			MAX

APは学習機能が使用できなくてもかまいません。また、常に2560バイトのワークが確保できるのであれば、このファンクションを呼ぶ必要はありません。

■ MSX-JE

このファンクションに限って、ワークエリアのアドレスはパラメータとしてMSX-JEに渡されません。したがって、MSX-JEはワークエリアを使用できません。

関連項目

Invoke（Function 2）

Invoke (Function 2)

機能

MSX-JE を起動します。

コール手順

A	02H
HL	ワークエリアのアドレス
DE	ワークエリアの大きさ

戻り値

なし

解説

AP で確保したワークエリアを初期化します。

注意

■ AP

AP は最低でも、Inquiry (Function 1) で返された MIN 以上のワークエリアが確保できなければなりまんせ。DE レジスタに MIN 以下の数を渡したときの結果は保証されません。

■ MSX-JE

MSX-JE は使用するワークエリアの大きさが一定ならば、このファンクションのパラメータは無視してかまいません。

関連項目

Inquiry (Function 1)

Release (Function 3)

機能

MSX-JE を終了します。

コール手順

A　　03H

HL　　ワークエリアのアドレス

戻り値

なし

解　説

APが確保したメモリを解放します。

注　意

■ AP

APはシステムの再起動や学習辞書の更新が保証されなくてもよければ、このファンクションを呼び出さなくてもかまいません。

■ MSX-JE

MSX-JEはこのファンクションによって、システムの再起動などに問題がなければ、なにもしなくてかまいません。

関連項目

Inquiry（Function 1）

Invoke（Function 2）

Clear（Function 4）

機　能

かな漢字変換用のバッファをクリアします。

コール手順

A　　04H

HL　　ワークエリアのアドレス

戻り値

なし

解 説

かな漢字変換用のバッファをクリアします。このファンクションにより、かな漢字変換モジュールはいままでの入力テキストをクリアします。

関連項目

Set TTB (Function 5)

Set TTB (Function 5) [option]

機 能

APからMSX-JEに再変換したいテキストと読みデータを渡し、MSX-JEの内部のバッファにセットします。

コール手順

A	05H
HL	ワークエリアのアドレス
DE	再変換するテキストデータのアドレス
BC	TTBのアドレス

戻り値

A	00H	再変換可能
	FFH	再変換不能

解 説

APからMSX-JEに再変換したいテキストと読みデータを渡し、MSX-JEの内部のバッファにセットします。このファンクションにより、MSX-JEは与えられたテキストを再変換します。

DEレジスタにテキストそのもののアドレスをセットし、BCレジスタにはTTB(Transferrable Text Block)のアドレスをセットします。APは再変換をする必要が無ければ、このファンクションを使用する必要はありません。MSX-JEがこのファンクションとGet TTB (Function 8) をサポートしない場合、A=0FFHとしてリターンします。

関連項目

5.4.5 TTB
Clear (Function 4)
Get TTB (Function 8)

Dispatch（Function 6）

機　能

APからMSX-JEにCPUの制御を渡します。

コール手順

A　　06H

HL　　ワークエリアのアドレス

戻り値

A　　ステータス（後述）

HL　　STBのアドレス

解　説

APからMSX-JEにCPUの制御を渡します。MSX-JEはテキスト表示の必要がある場合とAPに制御を戻す必要がある場合に、Aレジスタにステータスを、HLレジスタにSTBのアドレスをセットしてリターンします。MSX-JEが直接表示をすることは無く、表示するテキストはSTB（Screen image Text Block）の形でAPに渡され、APが表示することになります。

■ Dispatchのステータス

ビット	値	意　味
0	1	APがSTBの表示を行う。
1	1	APが変換結果を得られる。
2	1	MSX-JEの変換が終了した。

■取り得るステータスとその意味

ステータス	意　味
000	MSX-JEがキーを無視した場合とキー入力がなかった場合。
001	入力中もしくは変換中である場合。
01x	部分確定をした場合。
10x	変換が中断されて終了した場合。
11x	全確定した場合。

■ステータスの解釈とその順番

ステータスは3ビットの情報からなりますが、APはこれをそれぞれ別々に解釈します。例えば、ビット0〜2が同時に立った場合、APはまずSTBの表示を行い、次にGet Resultをコールし、Dispatchのコールを中止します。

■確定時のSTB

部分確定したときのSTBは、未確定部分を再表示する必要があります。この場合、変換結果を得る前と後とで学習機能などのため、表示内容が変わる可能性があります。そのため、AP側では、部分確定の変換結果を得た直後、再度Dispatchを呼び、表示内容に変更があったSTBを受け取らなければなりません。また、MSX-JE側は、部分確定の変換結果を返した直後にDispatchを呼び出された場合、キー処理を行わず、000または001のステータスを直ちに返します。

テールタイプ（Inquire Window Size参照）のウインドウの場合、入力処理を行うと、ウインドウ自体が移動し、以前の表示内容は保証されません。したがって、確定直後のSTBは表示テキストの全内容を含んでいます。

関連項目

5.4.4 STB

Get Result（Function 7）

機能

変換結果を返します。

コール手順

A　　07H

HL　　ワークエリアのアドレス

戻り値

HL　　得られた変換結果の先頭アドレス

解説

変換結果を返します。変換結果は00Hで終るシフトJIS文字列です。APは得られたテキストをあたかもコンソールから入力されたのと同じように処理します。

関連項目

Dispatch（Function 6）

Get TTB (Function 8)　[option]

機　能

Get Resultで得られたテキストの読みデータを獲得します。

コール手順

A　　08H
HL　　ワークエリアのアドレス

戻り値

HL　　TTBのアドレス

解　説

Get Resultで得られたテキストの読みデータを獲得します。読みデータはTTB形式で与えられます。APは再変換の必要がなければ、このファンクションをコールする必要はありません。MSX-JEはこのファンクションとSet TTB (Function 5) をサポートしない場合、HLレジスタに、1バイトのNULL (0) へのポインタをセットする。

関連項目

Set TTB (Function 5)
5.4.5 TTB

Inquire Window Size (Function 9) [option]

機　能

ウインドウ形式を設定します。

コール手順

A　　09H
HL　　ワークエリアのアドレス
E　　テール形式の最大長
B　　ウインドウ形式の最大高さ
C　　ウインドウ形式の最大長

戻り値

HL　　ウインドウ指定データのアドレス

解 説

MSX-JE用変換ウインドウをデフォルト（1行×全角14文字、インデペンデント形式）以外に設定します。まず、APが用意できるウインドウの大きさをパラメータとしてコールします。それに対して、MSX-JEは自分が必要とするウインドウの大きさをセットしてリターンします。デフォルトのウインドウサイズで使用するときは、APはこのファンクションを呼ぶ必要はありません。

■ **ウインドウ指定データ**

ウインドウ指定データ　　::= 任意個のウインドウ指定子，0
ウインドウ指定子　　　　::= ウインドウタイプ（1バイト）
　　　　　　　　　　　　　　ウインドウ長さ（1バイト）
　　　　　　　　　　　　　　ウインドウ高さ（1バイト）
ウインドウタイプ　　　　::= 1（インデペンデント）、2（テール）

文字数は、すべて半角相当の文字数です。ウインドウタイプは2種類あります。

ウインドウタイプ	機　能
インデペンデント	テキストと独立して折かえしの無いスクリーンイメージのウインドウです。
テール	APのテキストバッファの後に表示可能な（したがって折かえしやスクロールを許す）タイプのウインドウです。

注 意

■ **AP**

APがこのファンクションをコールしない場合、デフォルトのウインドウとしてインデペンデントタイプの全角14文字表示のウインドウを開きます。

■ **MSX-JE**

MSX-JEはこのファンクションをコールされなくても（デフォルトのウインドウのままで）動作しなければなりません。MSX-JEはデフォルトのウインドウ以外をサポートしない場合、このファンクションをコールされたら以下のデータの先頭アドレスをHLレジスタに入れてリターンして下さい（デフォルトのウインドウを再度指定）。

オフセット	データ
0	1
1	28
2	1
3	0

関連項目

5.4.4 STB

Conflict detect (Function 10) [option]

機　能

キー割り当ての衝突を防ぎます。

コール手順

A　　0AH
HL　　ワークエリアのアドレス

戻り値

A　　00H　　Not detected
　　　FFH　　Detected

解　説

APがキー入力キューから取りださずにMSX-JEに制御を移すキーとMSX-JEからAPに制御を移すキーが衝突することを防ぎます。
APは、デフォルトのキー割り当て(5.4.1参照)で、APがキューから取りださなくてはいけないキーをキューから取り出さずにMSX-JEに制御を渡したい場合、このファンクションをコールします。
MSX-JEは、もしキューの先頭の文字がDispatchに制御が渡ったときにキューから取りだされない文字だったら、このファンクションコールでキューから取りのぞき、A=0FFHとしてリターンします。そうでなければ、A=0としてリターンします。

注　意

■ MSX-JE
デフォルトのキー割り当てを使用するMSX-JEは、このファンクションでは常にキューから1文字取りだし、FFHを返します。

関連項目

5.4.1 キー割り当て

5.5 辞書インターフェイス

ここでは、MSX-JEの辞書アクセスインターフェイスについて解説します。辞書インターフェイスは日本語ワープロなど、高度なAPが使用するのに適したファンクションです。

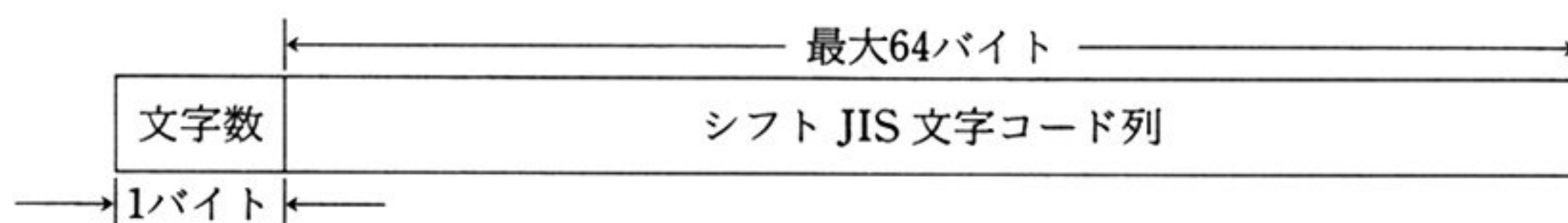
ゲームなど日本語入力を主体としないAPには、仮想端末入力インターフェイスが適しています。

ユーザーがAPごとに異なるユーザーインターフェイスを強いられ、困惑することを避けるためにも、特に必要のない限り、仮想端末インターフェイスを使用することを推奨します。

5.5.1 テキストデータ形式

辞書インターフェイスでは、テキストは常に以下のようなフォーマットで扱い、BCレジスタまたはDEレジスタで示されるバッファに置きます。

文字数	シフトJIS文字コード列

(文字数: 1バイト、シフトJIS文字コード列: 最大64バイト)

全角文字はシフトJISコードを用い、文字数は半角（バイト数）です。

5.5.2 ファンクション

表 7.98　辞書インターフェイス一覧

ファンクション番号	関数名	機　能
40H	han_zen	半角文字列を全角文字列に変換
41H	zen_han	全角文字列を半角文字列に変換
42H	han_kata	半角文字列を全角カタカナ文字列に変換（ローマ字変換）
43H	han_hira	半角文字列を全角ひらがな文字列に変換（ローマ字変換）
44H	kata_hira	全角カタカナ文字列を全角ひらがな文字列に変換
45H	hira_kata	全角ひらがな文字列を全角カタカナ文字列に変換
46H	open_dic	辞書ファイルのオープン
47H	henkan	連文節変換の開始
48H	ji_koho	次候補の獲得
49H	zen_koho	前候補の獲得
4AH	ji_block	次候補ブロックの獲得
4BH	zen_block	前候補ブロックの獲得
4CH	kakutei1	候補全体の確定
4DH	kakutei2	先頭文節の確定
4EH	close_dic	辞書ファイルのクローズ
4FH	touroku	単語登録
50H	sakujo	単語削除

han_zen (Function 40H)

機　能

半角文字列を全角文字列に変換します。

コール手順

A	40H
HL	ワークエリアのアドレス
DE	ソースバッファ（半角文字列）のアドレス
BC	デスティネーション（全角文字列）のアドレス

戻り値

A	00H	成功
	0以外	失敗、変換できなかった文字のソース上の位置

解　説

半角文字列を全角文字列に変換します。半角コード00H〜1FH、7FH〜9FH、E0H〜FFHはエラーになります。戻り値として、成功すれば0、失敗すれば失敗した文字のソースバッファ上での位置が返ります。失敗の場合も含めて、デスティネーションには変換された文字数と文字列をセットします。
ただし、半角記号、濁音、半濁音記号の取り扱いについては、処理系によって変換結果が異なることがあります。

例

以下のプログラムは半角文字列「ABC」を全角文字列「ＡＢＣ」に変換する。

■**プログラム**

```
#define han_zen(s, d) mjea((char)0x40, s, d)
char    s[65], d[65];
        han_zen(s, d);
```

■**入力データ**

s	3	ABC	(半角文字列)

■**実行結果**

d	6	ＡＢＣ	(全角文字列)

zen_han (Function 41H)

機　能

全角文字列を半角文字列に変換します。

コール手順

A　　41H
HL　　ワークエリアのアドレス
DE　　ソースバッファ（全角文字列）のアドレス
BC　　デスティネーション（半角文字列）のアドレス

戻り値

A　　00H　　成功
　　0以外　　失敗、変換できなかった文字のバッファ上での位置

解　説

全角文字列を半角文字列に変換します。半角文字と対応する半角文字のない全角文字はエラーになります。戻り値として、成功すれば0、失敗すれば失敗した文字のソースバッファ上での位置が返ります。失敗の場合も含めて、デスティネーションには変換された文字数と文字列をセットします。
ただし、全角記号、ひらがな、カタカナの濁音、半濁音文字については処理系によって変換結果が異なることがあります。

例

以下のプログラムは全角文字列「ＡＢＣ」を半角文字列「ABC」に変換する。

■プログラム

```
#define  zen_han(s, d)  mjea((char)0x41, s, d)
char     s[65], d[65];
         zen_han(s, d);
```

■入力データ

s	6	ＡＢＣ	（全角文字列）

■実行結果

d	3	ABC	（半角文字列）

han_kata (Function 42H)

機　能

ローマ字、カタカナ、またはその混在の半角文字列を全角カタカナ文字列に変換します。

コール手順

A	42H
HL	ワークエリアのアドレス
DE	ソースバッファ（半角文字列）のアドレス
BC	デスティネーション（カタカナ文字列）のアドレス

戻り値

A	00H	成功
	0以外	失敗、変換できなかった文字のバッファ上での位置

解　説

ローマ字、カタカナ、またはその混在の半角文字列を全角カタカナ文字列に変換します。戻り値として、成功すれば0、失敗すれば失敗した文字のソースバッファ上での位置が返ります。失敗の場合も含めて、デスティネーションには変換された文字数と文字列をセットします。

例

以下のプログラムは半角文字列「katakana」を全角カタカナ文字列「カタカナ」に変換する。

■**プログラム**

```
#define han_kata(s, d) mjea((char)0x42, s, d)
char     s[65], d[65];
         han_kata(s, d);
```

■**入力データ**

s	8	katakana	（半角文字列）

■**実行結果**

d	8	カタカナ	（全角文字列）

han_hira（Function 43H）

機　能

ローマ字、カタカナ、またはその混在の半角文字列を全角ひらがな文字列に変換します。

コール手順

A	43H
HL	ワークエリアのアドレス
DE	ソースバッファ（半角文字列）のアドレス
BC	デスティネーション（ひらがな文字列）のアドレス

戻り値

A	00H	成功
	0以外	失敗、変換できなかった文字のバッファ上での位置

解　説

ローマ字、カタカナ、またはその混在の半角文字列を全角ひらがな文字列に変換します。戻り値として、成功すれば0、失敗すれば失敗した文字のソースバッファ上での位置が返ります。失敗の場合も含めて、デスティネーションには変換された文字数および文字列をセットします。

例

以下のプログラムは半角文字列「hiragana」を全角ひらがな文字列「ひらがな」に変換する。

■プログラム

```
#define han_hira(s, d) mjea((char)0x43, s, d)
char    s[65], d[65];
        han_hira(s, d);
```

■入力データ

s	8	hiragana	(半角文字列)

■実行結果

d	8	ひらがな	(全角文字列)

kata_hira（Function 44H）

機　能

全角カタカナ文字列を全角ひらがな文字列に変換します。

コール手順

A	44H
HL	ワークエリアのアドレス
DE	ソースバッファ（カタカナ文字列）のアドレス
BC	デスティネーション（ひらがな文字列）のアドレス

戻り値

なし

解　説

全角カタカナ文字列を全角ひらがな文字列に変換します。全角カタカナ文字以外の全角文字はそのままデスティネーションにコピーします。半角文字が含まれていた場合の変換結果は保証しません。
ただし、「ヴ」、「ヵ」、「ヶ」については処理系によって変換結果が異なることがあります。

例

以下のプログラムは全角カタカナ文字列「カタヒラ」を全角ひらがな文字列「かたひら」に変換する。

■プログラム

```
#define  kata_hira(s, d) mjea((char)0x44, s, d)
char     s[65], d[65];
         kata_hira(s, d);
```

■入力データ

s	8	カタヒラ	（全角文字列）

■実行結果

d	8	かたひら	（全角文字列）

hira_kata（Function 45H）

機　能

全角ひらがな文字列を全角カタカナ文字列に変換します。

コール手順

A	45H
HL	ワークエリアのアドレス
DE	ソースバッファ（ひらがな文字列）のアドレス
BC	デスティネーション（カタカナ文字列）のアドレス

戻り値

なし

解　説

全角ひらがな文字列を全角カタカナ文字列に変換します。全角ひらがな文字以外の全角文字はそのままデスティネーションにコピーします。半角文字が含まれていたときの変換結果は保証しません。

例

以下のプログラムは全角ひらがな文字列「ひらかた」を全角カタカナ文字列「ヒラカタ」に変換します。

■プログラム

```
#define hira_kata(s, d) mjea((char)0x45, s, d)
char    s[65], d[65];
        hira_kata(s, d);
```

■入力データ

s	8	ひらかた	(全角文字列)

■実行結果

d	8	ヒラカタ	(全角文字列)

open_dic（Function 46H）

機 能

将来の拡張用です。

コール手順

A	46H
HL	ワークエリアのアドレス
DE	0

戻り値

A	05H（常に5を返す）

解 説

この機能は将来の拡張のために用意されています。アプリケーションはこの機能を呼ぶ必要はありません。

henkan（Function 47H）

機 能

コール手順

A	47H
HL	ワークエリアのアドレス
DE	読みデータ（カタカナ文字列）のアドレス

戻り値

A	00H	候補なし
	0以外	候補の数

解 説

全角カタカナ、全角ひらがな文字列を漢字かな混じり文字列に変換します。戻り値として、先頭文節の候補数が返ります。変換結果は、ji_kouho（Function 48H）などで取り出すことができます。

例

以下のプログラムは文字列tに入っている読みデータを漢字かな混じり文字列に変換します。

■プログラム

```
#define  henkan(s)  mjea((char)0x47, s)
char     t[65];  TINY    n;
         n = henkan(t);
```

■入力データ

t	10	ヤマノウエ	(全角文字列)

■実行結果

nに先頭文節「ヤマノ」の候補数をセットされます。変換結果はji_kohoなどで得ることができます。

ji_koho (Function 48H)

機能

先頭文節の候補群の中の次の候補を1つ獲得します。

コール手順

A	48H
HL	ワークエリアのアドレス
DE	候補バッファ
BC	後続文節バッファ

戻り値

A	00H	候補なし
	0以外	候補番号

解説

先頭文節の候補群の中の次の候補を1つ獲得します。この呼び出しの前に、henkan (Function 47H) で読みデータをセットしなければなりません。

例

以下のプログラムは「ヤマノウエノイエ」を変換して、「山の」を cand に、「上の家」を rest にセットします。

■**プログラム**

```
#define ji_koho(c, r) mjea((char)0x48, c, r)
char    text[65], cand[65], rest[65];
        henkan(text);
        ji_koho(cand, rest);
```

■**入力データ**

text	16	ヤマノウエノイエ	(全角文字列)

■**実行結果**

cand	4	山の	(全角文字列)
rest	6	上の家	(全角文字列)

zen_koho (Function 49H)

機 能

先頭文節の候補群の中の1つ前の候補を獲得します。

コール手順

A	49H
HL	ワークエリアのアドレス
DE	候補バッファ
BC	後続文節バッファ

戻り値

A	00H	候補なし
	0以外	候補番号

解 説

先頭文節の候補群の中の1つ前の候補を獲得します。前もって、henkan(Function 47H)で読みデータをセットしなければなりません。

例

以下のプログラムは「キカイヲミテ」を「機械を・・・」、「機会を・・・」に変換したあと、第一候補の「機械を・・・」を再度獲得します。

■ **プログラム**

```
#define  zen_koho(c, r) mjea((char)0x49, c, r)
char     text[65], cand[65], rest[65];
         henkan(text);
         ji_koho(cand, rest);
         ji_koho(cand, rest);
         zen_koho(cand, rest);
```

■ **入力データ**

text	12	キカイヲミテ	(全角文字列)

■ **実行結果**

1回目の ji_koho の結果

cand	6	機械を	(全角文字列)
rest	4	見て	(全角文字列)

2回目の ji_koho の結果

cand	6	機会を	(全角文字列)
rest	4	見て	(全角文字列)

zen_koho の結果

cand	6	機械を	(全角文字列)
rest	4	見て	(全角文字列)

ji_block (Function 4AH)

機　能

現在の候補群の次に優先度の低い候補群を作成し、候補数を返します。

コール手順

A　4AH

HL　ワークエリアのアドレス

戻り値

A　00H　ブロック無し

0以外　候補の数

解　説

現在の候補群の次に優先度の低い候補群を作成し、候補数を返します。前もって、henkan (Function 47H) で読みデータをセットしなければなりません。

いったん読みデータをセットすれば、いつ、何回、ji_block を行なってもかまいません。結果は ji_koho などで獲得します。

例

以下のプログラムは「キョウハイシャニイッタ」を変換します。最初の ji_koho の結果は「今日は」と「医者にいった」になります。次に、ji_block で次の候補群を作成すると、ji_koho の結果、「今日」と「歯医者にいった」が獲得されます。

■**プログラム**

```
#define  ji_block()  mjea((char)0x4A)
char     text[65], cand[65], rest[65];
         henkan(text);
         ji_koho(cand, rest);
         ji_block();
         ji_koho(cand, rest);
```

■**入力データ**

text	22	キョウハイシャニイッタ	(全角文字列)

■**実行結果**

1回目の ji_koho

cand	6	今日は	(全角文字列)
rest	12	医者にいった	(全角文字列)

2回目の ji_koho

cand	4	今日	(全角文字列)
rest	14	歯医者にいった	(全角文字列)

zen_block (Function 4BH)

機　能

コール手順

A　　4BH
HL　　ワークエリアのアドレス

戻り値

A　　00H　　ブロック無し
　　0以外　候補の数

解　説

現在の候補群より優先度の高い候補群を作成し候補数を返します。前もって、henkan (Function 47H) で読みデータをセットしなければなりません。
zen_block の使用方法は、ji_block と同じです。いったん読みデータをセットすれば、いつ、何回、zen_block を実行してもかまいません。結果は、ji_koho などで獲得します。

例

以下のプログラムは、いったん ji_block をして、再度 zen_block を行ってそれぞれ変換結果を得ます。

■プログラム

```
#define  zen_block()  mjea((char)0x4b)
char     text[65], cand[65], rest[65];
         henkan(text);
         ji_koho(cand, rest);
         ji_block();
         ji_koho(cand, rest);
         zen_block();
         ji_koho(cand, rest);
```

■入力データ

text | 22 | キョウハイシャニイッタ | (全角文字列)

■実行結果

1回目の ji_koho の結果

cand | 6 | 今日は | (全角文字列)

rest | 12 | 医者にいった | (全角文字列)

2回目の ji_koho の結果

cand | 4 | 今日 | (全角文字列)

rest | 14 | 歯医者にいった | (全角文字列)

3回目の ji_koho の結果

cand | 6 | 今日は | (全角文字列)

rest | 12 | 医者にいった | (全角文字列)

kakutei1（Function 4CH）

機　能

変換結果を確定します。

コール手順

A　4CH
HL　ワークエリアのアドレス
E　候補ブロック内通し番号
BC　変換結果格納バッファアドレス

戻り値

なし

解　説

先頭文節として、Eレジスタで指定された候補を採用して、文全体を変換し、結果を指定されたバッファに格納します。

前もってhenkan(Function 47H)で読みデータをセットしなければなりません。ji_koho (Function 48H) などは必ずしも必要ありません。

例

以下のプログラムは、sに入っている読みデータを変換してtに格納します。先頭文節の候補として3番目の候補を使います。

■プログラム

```
#define  kakutei1(c, s)   mje((char)0x4c, c, s)
char     s[65], t[65];
         henkan(s);
         kakutei1(3, t);
```

■入力データ

s	14	キシャノキシャ	(全角文字列)

■実行結果

t	10	貴社の記者	(全角文字列)

kakutei2 (Function 4DH)

機　能

コール手順

A	4DH
HL	ワークエリアのアドレス
E	候補ブロック内通し番号
BC	変換結果格納バッファアドレス

戻り値

A	確定した先頭文節の読みの長さ

解　説

先頭文節として、Eレジスタで指定された候補を採用して、先頭文節を変換し、結果を指定されたバッファに格納します。読みデータのなかで、先頭の文節として変換された部

分の長さ（半角相当）をAレジスタに返します。
前もってhenkan（Function 47H）で読みデータをセットしておくことが必要です。ji_koho などは必ずしも必要ではありません。

例

以下のプログラムはsに入っている読みデータの最初の文節を変換してtに格納します。先頭文節の候補として最初の候補を用います。次に、この関数の戻り値を用いて後続文節の読みデータの先頭を求め、後続文節を新たに変換します。

■プログラム

```
#define  kakutei2(c, s) mje((char)0x4d, c, s)
char     s[65], t[65], *u;
         henkan(s);
         n = kakutei2(1, t);
         u = s+n;
         *u = *s-n;
         henkan(u);
```

■入力データ

s　| 14 | キシャノキシャ |　(全角文字列)

■実行結果

n = 8

t　| 6 | 貴社の |　(全角文字列)

|←―― 8バイト ――→|

s　| 14 | キシャ・・・ | 6 | キシャ |　(全角文字列)

u ――→ (6 を指す)

close_dic（Function 4EH）

機　能

ダミーのファンクションで、何もしません。

コール手順

A　　4EH
HL　　ワークエリアのアドレス

戻り値

A　　00H（常に0）

解　説

この関数はVJEとの互換のためにあり、内部では何もしません。

touroku（Function 4FH）

機　能

読みデータ、単語データ、品詞を与えて、指定単語を辞書に登録します。

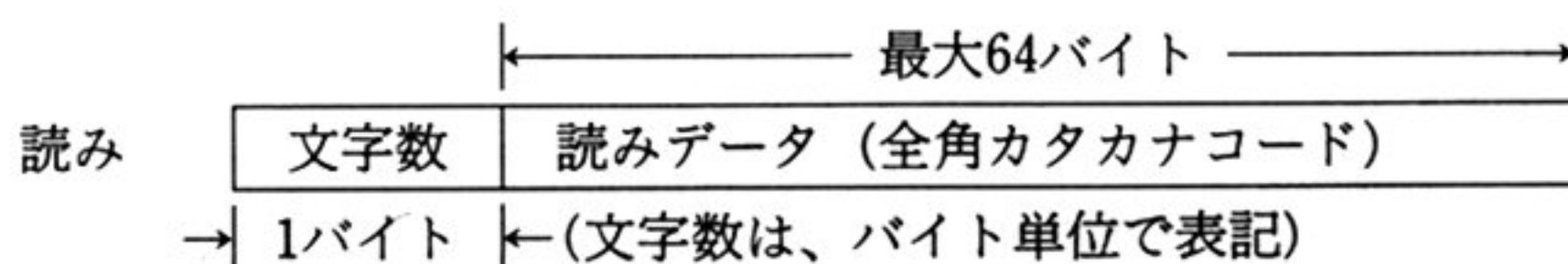
コール手順

AT	4FH
HL	ワークエリアのアドレス
DE	読みバッファのアドレス
BC	登録単語バッファのアドレス

戻り値

A	00H	正常終了（登録された）
	01H	空き領域が無い
	02H	同音語数オーバーフロー
	04H	読みデータが不適切
	08H	単語データが不適切
	10H	品詞情報が不適切
	FFH	サポートされていない

解　説

読みデータ、単語データ、品詞を与えて、指定単語を辞書に登録します。読みバッファ・登録単語バッファ共に、以下のように先頭に文字数を持つ文字列の形式です。品詞は、登録単語バッファの直後の1バイトに格納して引き渡します。

	←1バイト→	←最大64バイト→
読み	文字数	読みデータ（全角カタカナコード）

（文字数は、バイト単位で表記）

	最大64バイト		
単語	文字数	単語データ（全角文字コード）	品詞
	1バイト	（文字数は、バイト単位で表記）	1バイト

読みデータ

2バイト以上64バイト以下の全角カタカナコードです。ただし、「ヮ」、「ヰ」、「ヱ」、「ヴ」、「ヵ」、「ヶ」は、処理系によって登録・削除できない可能性があります。

単語データ

2バイト以上64バイト以下の全角文字コードです。

品詞コード

以下に示すコード以外はエラーになります。

コード	意　味
1	名詞
2	名（姓）
3	名（名）
4	名
5	サ変動詞（名詞＋する）

例

以下のプログラムは、読みが「ミツオ」で品詞が人名（名）である単語「光雄」を登録します。

■ **プログラム**

```
#define  touroku(y, t) mje((char)0x4F, y, t)
char     yomi[65], tan[66];
         touroku(yomi, tan);
```

■ **入力データ**

	←6バイト→	
yomi	6	ミツオ（全角文字列）

	←4バイト→		
tan	4	光雄（全角文字列）	3

■ **実行結果**

n＝0（正常終了）

読みが「ミツオ」で品詞が人名（名）である「光雄」が登録されます。

sakujo（Function 50H）

機　能

コール手順

A	50H
HL	ワークエリアのアドレス
DE	読みバッファのアドレス
BC	削除単語バッファのアドレス

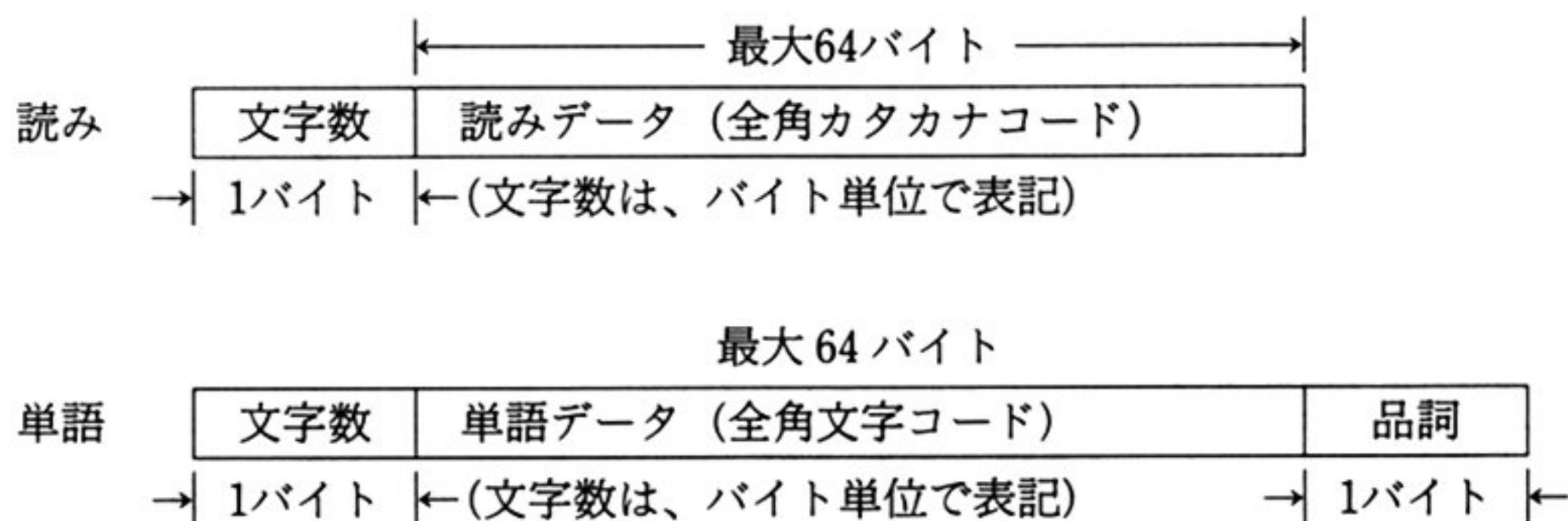
戻り値

A	00H	正常終了（削除された）
	01H	削除対象の単語がみつからない
	02H	（未使用）
	04H	読みデータが不適切
	08H	単語データが不適切
	10H	品詞情報が不適切
	FFH	サポートされていない

解　説

読みデータ、単語データ、品詞を与えて、指定単語を辞書から削除します。読みバッファ・削除単語バッファ共に、以下のように先頭に文字数を持つ文字列の形式です。品詞は、削除単語バッファの直後の1バイトに格納して引き渡します。

		最大64バイト
読み	文字数	読みデータ（全角カタカナコード）
	1バイト	←（文字数は、バイト単位で表記）

		最大64バイト	
単語	文字数	単語データ（全角文字コード）	品詞
	1バイト	←（文字数は、バイト単位で表記）	1バイト

読みデータ

2バイト以上64バイト以下の全角カタカナコードです。ただし、「ヮ」、「ヰ」、「ヱ」、「ヴ」、「ヵ」、「ヶ」については、FEPの種類やバージョンによって登録・削除できない可能性があります。

単語データ

2バイト以上64バイト以下の全角文字コードです。

品詞コード

以下に示すコード以外はエラーになります。

コード	意　味
1	名詞
2	人名（姓）
3	人名（名）
4	地名
5	サ変動詞（名詞＋する）

例

以下のプログラムは、読みが「ミツオ」で品詞が人名（名）である単語「光雄」を削除します。

■プログラム

```
#define  sakujo(y, t) mje((char)0x50, y, t)
char     yomi[65], tan[66];
         sakujo(yomi, tan);
```

■入力データ

		6バイト
yomi	6	ミツオ（全角文字列）

		4バイト	
tan	4	光雄（全角文字列）	3

■実行結果

n＝0（正常終了）

読みが「ミツオ」で品詞が人名（名）である「光雄」が削除されます。

5.6 VJE-80 および VJE-80A 使用上の注意

ここでは、アスキーが開発した VJE-80(日本語 MSX-Write に搭載)、VJE-80A(日本語 MSX-Write2 に搭載)を使用する上で、特に注意しなければならないことを解説します。VJE-80 および VJE-80A の仕様については、各仕様書をご覧下さい。

5.6.1 MSX-JE の起動

1. 実装の確認

MSX-JE を使用する AP は、必要とするインターフェイスを持つ MSX-JE デバイスが MSX に実装されているかを確認しなければなりません。

もし、必要とするデバイスが実装されていない場合は、日本語入力ができないことをユーザーに知らせ処理を終了するか、もしくは他の方法で日本語処理をしなければなりません。例えば、単漢字変換機能を AP 側で持つなどが考えられます。

MSX-JE のアクセス方法は、「5.3.1 エントリアドレスの獲得」をご参照下さい。

エントリアドレスの獲得ファンクションによる、HL レジスタの値およびケーパビリティベクトルの内容から自分が使用する MSX-JE の有無、仕様などを判断してください。

具体的には次のようにします。

1. HL レジスタの内容からテーブルが作成されたことを確認します。
   指定デバイスが実装されていれば HL レジスタの内容が更新されています。
2. ケーパビリティベクトルの内容から仕様を判断します。
   1.、2. いずれの場合も使用不可能ならばエラー処理して下さい。
3. MSX-JE のケーパビリティベクトルは図 7.50 のようになっています。

各ビットから自分が使用できるものかを判断してください。

仮想端末インターフェイスのみを使用するときは、以下のようにします。

```
ld      a, ケーパビリティベクトル
and     03h
jnz     エラー処理エントリ
```

辞書インターフェイスのみを使用するときは、以下のようにします。

```
ld      a, ケーパビリティベクトル
and     05h
jnz     エラー処理エントリ
```

辞書、仮想端末インターフェイスの両者を使用するときは、以下のようにします。

```
ld      a, ケーパビリティベクトル
and     07h
jnz     エラー処理エントリ
```

注　意

1. VJE-80のケーパビリティベクトルはF8Hですが、他のFEPがそうなっているとは限りません。商用のAPはケーパビリティベクトルとF8Hとのコンペアをとるなどの方法で、MSX-JEの有無や仕様を判断してはいけません。
2. MSX-JE仕様では単語・登録削除機能は必須ではないので、AP側は登録削除機能が無い場合も考慮しなければなりません。

2. 2つのMSX-JEが実装されていた場合

一部のMSX2、MSX2＋にはMSX-JEが内蔵されており、これは切り放すことはできません。また、2つのバージョンの異なるMSX-JEが実装されることが考えられます。

MSX-JEを使用するアプリケーションは拡張BIOSコールにより作成されたデバイス情報テーブルから、これらの情報を知ることができるので、実装されている全てのMSX-JEのケーパビリティベクトルをチェックしてMSX-JEを使用して下さい。

3. MSX-JEのファンクションコール

辞書インターフェイス、仮想端末インターフェイスの区別なく、CPUの各レジスタに値を設定したうえで、先に獲得したMSX-JEのスロットアドレス、エントリアドレスをインタースロットコールします。

5.6.2 仮想端末インターフェイスを使用するときの注意点

以下に、仮想端末インターフェイスを使うときの注意を説明します。

1. ウインドウサイズ

仮想端末インターフェイスを使う場合には、inquire window size をコールしてウインドウサイズを明示的にすることを推奨します。この関数をコールしない場合、MSX-JE は常にデフォルトのウインドウを開きますが、ウインドウサイズはできるだけ大きい方が使い勝手がよくなることが多いためです。

2. STB の表示

MSX で漢字を表示するときはビットマップ表示になるので、STB の表示方法次第で使用感が大きく異なります。以前表示されていた STB と同じ部分はなるべく表示しないなどの工夫をして、表示速度を上げる努力をして下さい。

3. 仮想端末のキーアサイン

仮想端末インターフェイスのキーアサインは MSX-JE 仕様で規定されていないので、FEP 間で異なります。

表 7.99　VJE-80A のキーアサイン

キー	動　作
[SPACE]	文節変換および変換中の次候補（次の同音異義語）の表示。
[SHIFT]+[SPACE]	前候補(同一ブロック内のみ有効、変換中のみ機能する。変換中以外は空白1文字)。
[F1]	ローマ字入力モードと英数字入力モードの切り換え。
[F2]	ひらがな変換（対象文節をひらがなに変換する）。
[F3]	カタカナ変換（対象文節をカタカナに変換する）。
[F4]	半角変換(英数字とカタカナのみ、ひらがな・漢字も半角カタカナに変換される。文節単位に変換することはできない。半角変換では、カーソル位置までが対象となる)。
[F5]	コード変換（入力された JIS コードに相当する単漢字に変換する）。
[CTRL]+[F5]	全文確定。
[↓]	文節変換／次の同音異義語のブロックの先頭を表示。
[↑]	文節変換／前の同音異義語のブロックの先頭を表示。
[←]	カーソル左移動。
[→]	カーソル右移動。

キー	動　作
任意の1文字	全文確定(確定キーを押さなくても、次の文字を入力すると、それ以前に変換された内容が確定する)。
[SHIFT]+[←]	カーソルを入力行頭へ移動する。
[SHIFT]+[→]	カーソルを入力行末へ移動する。
[ESC]	処理の取消（変換後→変換前→取消)。
[RETURN]	文節単位の確定。
[BS]	カーソル左の1文字を消す。(変換中に押した場合は全確定し、最後の1文字が消去される)。
[DEL]	カーソル上の1文字を消す。

5.6.3 VJE-80とVJE-80Aとの互換性

MSX-JEの仕様を遵守している限り、互換性は保証します。

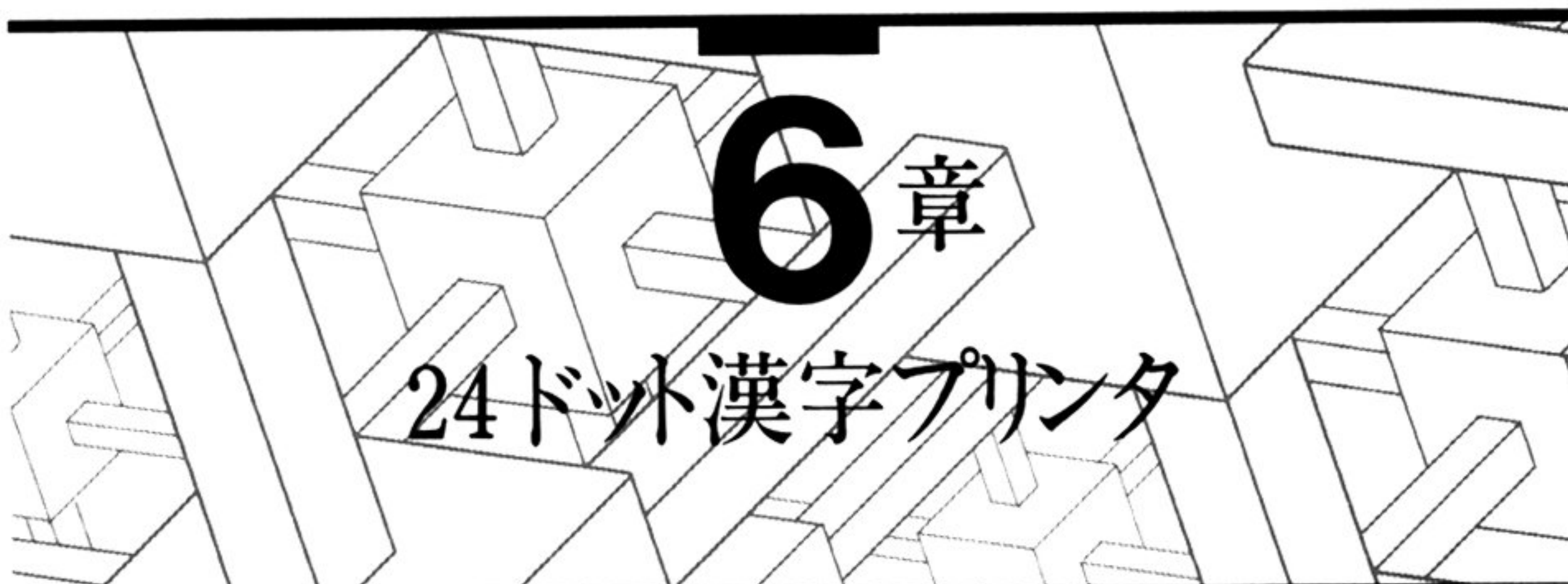
ただし、VJE-80Aのサポートする関数はVJE-80よりも増えています。また、ワークエリアの大きさも異なります。

このため、VJE-80Aで動作したAPが必ずVJE-80で動作するとは限りません。特に、VJE-80Aだけがサポートする関数を使った場合は、MSX-JE間での互換性も取れなくなりますのでご注意下さい。

なお、SRAM上の辞書および学習情報は、クローズしないで電源OFFまたはリセットされると壊れます。必ず、releaseによってメモリを解放して、辞書をクローズして下さい。

5.6.4 VJE-80およびVJE-80Aでの禁止事項

VJE-80および80Aを使用する場合に絶対行ってはならないことを記しておきます。この内容は通常起こらないはずですが、デバッグ中などに原因不明の症状が起こったとき、調べてみて下さい。

1. VJE-80およびVJE-80Aを含むスロットのページ1の任意アドレスにデータを書き込むことは禁止します。これは、VJE-80およびVJE-80Aがバンク構造で動いており、そのバンクセレクタがこの領域にマッピングされているためです。
2. VJE-80Aではページ2の後半（A000H～BFFFH）にワークエリアおよびユーザー辞書、学習情報などを保存しているSRAMがマッピングされます。したがって、この領域に対して書き込みを行ったときの動作保証はできません。

6章 24ドット漢字プリンタ

6.1 基本仕様

6.1.1 ドット構成

1. 縦ピッチ　　1/180 インチ
2. 横ピッチ　　1/180 インチ

6.1.2 TYPE A と TYPE B の違い

MSX24 ドット漢字プリンタには TYPE A と TYPE B の 2 種類の仕様があります。両者の違いはサポートするコントロールコードの数で、TYPE B に比べて TYPE A の方が多くのコントロールコードをサポートしています。TYPE A はフルセット版、TYPE B はサブセット版とお考えください。

コントロールコードの中には TYPE A、TYPE B とも必須でないものがあります。これらのコードをサポートする必要はありませんが、シーケンスはその機能に予約されています。すでに発売されているプリンタの中にはこれらオプションの機能を持ったものがあります。同じ機能をサポートする場合には、そのコードにしたがってください。また、新たに特別な機能を付加するときには、この仕様書に記載されているコードは割り当てないでください。また、別のコードを割り当てるときには、事前に当社までご一報ください。

6.2 コントロールコード

6.2.1 コントロールコード一覧

1. 制御コード表

シンボル	HEX コード	機　能	TYPE A	TYPE B
CR	OD	印字指令であり、印字後復帰	○	○
LF	OA	1行改行	○	○
VT	OB	垂直タブ位置まで改行	○	×
FF	OC	改ページ	○	○
SO	OE	拡大指令	○	○
SI	OF	拡大解除	○	○
HT	09	水平タブ位置へ移動	○	×
CAN	18	データをキャンセル	○	×
GS	1D	VFU のセット開始	○	×
RS	1E	VFU のセット終了	○	×
US	1F	VFU の実行または 1〜15 行改行	○	×
EOT	04	外字の登録データ終了	○	○
SOH	01	グラフィックキャラクターの指定	○	○

2. 拡張制御コード表（ESC シーケンス）

シンボル	HEX コード	機　能	TYPE A	TYPE B
ESC N	1B, 4E	パイカモードの設定	○	○
ESC H	1B, 48	パイカモードの設定	○	○
ESC Q	1B, 51	コンデンスモードの設定	○	×
ESC E	1B, 45	エリートモードの設定	○	×
ESC P	1B, 50	プロポーショナルモードの設定	○	×
ESC K	1B, 4B	漢字モード（横印字）の設定	○	○
ESC t	1B, 74	漢字モード（縦印字）の設定	○	○
ESC (01)	1B, 01	1ドットスペース	○	○
ESC (02)	1B, 02	2ドットスペース	○	○
ESC (03)	1B, 03	3ドットスペース	○	○
ESC (04)	1B, 04	4ドットスペース	○	○
ESC (05)	1B, 05	5ドットスペース	○	○

シンボル	HEX コード	機　能	TYPE A	TYPE B
ESC (06)	1B, 06	6ドットスペース	○	○
ESC (07)	1B, 07	7ドットスペース	○	○
ESC (08)	1B, 08	8ドットスペース	○	○
ESC S	1B, 53	8ドットビットイメージモードの設定	○	○
ESC I	1B, 49	16ドットビットイメージモードの設定	○	○
ESC J	1B, 4A	24ドットビットイメージモードの設定	○	○
ESC V	1B, 56	8ドットビットイメージリピート	○	×
ESC W	1B, 57	16ドットビットイメージリピート	○	×
ESC U	1B, 55	24ドットビットイメージリピート	○	○
ESC F	1B, 46	ドットアドレッシング	○	○
ESC A	1B, 41	1/6インチ改行モードの設定	○	○
ESC B	1B, 42	1/8インチ改行モードの設定	○	○
ESC T	1B, 54	N/120またはN/180インチ改行モードの設定	○	○
ESC (	1B, 28	水平タブセット	○	×
ESC)	1B, 29	水平タブ部分クリア	○	×
ESC 2	1B, 32	水平タブオールクリア	○	×
ESC X	1B, 58	アンダーラインの設定	○	○
ESC Y	1B, 59	アンダーラインの解除	○	○
ESC L	1B, 4C	印字開始桁の設定	○	○
ESC	1B, 2F	印字終了桁の設定	×	×
ESC *	1B, 2A	外字のロード (16 * 16ドット)	×	×
ESC +	1B, 2B	外字のロード (24 * 24ドット)	○	○
ESC D	1B, 44	コピーモード	○	○
ESC M	1B, 4D	ネイティブモード	○	×
ESC R	1B, 52	キャラクタリピート	○	×
ESC !	1B, 21	強調印字モード指定	○	○
ESC "	1B, 22	強調印字モード解除	○	○
ESC >	1B, 3E	片方向印字モード指定	○	○
ESC]	1B, 5D	片方向印字モード解除	○	○
ESC f	1B, 66	順方向改行モード	○	×
ESC r	1B, 72	逆方向改行モード	○	×
ESC a	1B, 61	用紙排出後給紙	×	×
ESC b	1B, 62	用紙排出	×	×
ESC C S	1B, 43, 53	スーパースクリプトモード	○	○
ESC C s	1B, 43, 73	サブスクリプトモード	○	○
ESC h 0	1B, 68, 30	半角文字縦印字解除	×	×
ESC q	1B, 71	半角組文字縦書指定	×	×
ESC e n1 n2	1B, 65	文字拡大率指定	×	×
ESC d n	1B, 64	n=1 ドラフト指定、n=0 ドラフト解除	×	×
ESC c 1	1B, 63, 31	ソフトウェアリセット	×	×
ESC C J	1B, 43, 4A	JISモードの指定	×	×

シンボル	HEXコード	機　能	TYPE A	TYPE B
ESC C j	1B, 43, 6A	JISモードの解除	×	×
ESC @	1B, 40	紙送りのバックラッシュ除去	×	×
ESC x	1B, 78	漢字フォントの転送（割り込みなし）	×	×
ESC y	1B, 79	漢字フォントの転送（割り込み有り）	×	×
ESC h 1	1B, 68, 31	半角文字縦印字指定	×	×
ESC C a	1B, 43, 61	A4用紙の印字	×	×
ESC C b	1B, 43, 62	B4用紙の印字	×	×
ESC C t	1B, 43, 74	葉書縦挿入時の紙送り初期化	×	×
ESC C y	1B, 43, 79	葉書横挿入時の紙送り初期化	×	×
ESC C T	1B, 43, 54	テスト印字モード	×	×
ESC p `	1B, 70, 60	漢字モードで英数記号の印字を標準書体に設定	×	×
ESC p a	1B, 70, 61	漢字モードで英数記号の印字をポエム体に設定	×	×
ESC p b	1B, 70, 62	漢字モードで英数記号の印字をデコ体に設定	×	×
ESC p c	1B, 70, 63	漢字モードでの英数記号の印字をマルチフォント化するために予約	×	×
∫	∫		×	×
ESC p	1B, 70, 7E		×	×
ESC p @	1B, 70, 40	漢字モードで平、片仮名の印字を標準書体に設定	×	×
ESC p A	1B, 70, 41	漢字モードで平、片仮名の印字をポエム体に設定	×	×
ESC p B	1B, 70, 42	漢字モードで平、片仮名の印字をかしこ体に設定	×	×
ESC p C	1B, 70, 43	漢字モードで平、片仮名の印字をマルチフォント化するために予約	×	×
∫	∫		×	×
ESC p _	1B, 70, 5F		×	×
ESC p Z	1B, 70, 5A	漢字モードでの印字をすべて標準書体に設定	×	×
ESC p 0	1B, 70, 30	文字飾り印字の解除	×	×
ESC p 1	1B, 70, 31	白抜き印字の設定	×	×
ESC p 2	1B, 70, 32	斜体印字の設定	×	×
ESC p 3	1B, 70, 33	太字印字の設定	×	×
ESC p 4	1B, 70, 34	文字飾り印字に予約	×	×
∫	∫		×	×
ESC p ?	1B, 70, 3F		×	×
ESC p !	1B, 70, 21	文字飾り印字に予約	×	×
∫	∫		×	×
ESC p /	1B, 70, 2F		×	×

3. その他拡張制御コード（SUBシーケンス）

シンボル	HEXコード	機　能	TYPE A	TYPE B
SUB A	1A, 41	960ドット/8インチ基本ピッチ	×	×
SUB B	1A, 42	1280ドット/8インチ基本ピッチ	×	×
SUB C	1A, 43	1440ドット/8インチ基本ピッチ	×	×
SUB D	1A, 44	720ドット/8インチ基本ピッチ	×	×
SUB F	1A, 46	1/120インチ改行ピッチ	○	×
SUB G	1A, 47	1/180インチ改行ピッチ	○	×
SUB 1	1A, 31	外字登録部分クリア	×	×
SUB 2	1A, 32	外字登録オールクリア	○	×
SUB U	1A, 55	スーパースクリプトモード	○	×
SUB L	1A, 4C	サブスクリプトモード	○	×
SUB V	1A, 56	縦拡大文字モードの設定	○	×
SUB W	1A, 57	縦拡大文字モードの解除	○	×
SUB Q	1A, 51	漢字24ドットピッチ	×	×
SUB N	1A, 4E	漢字27ドットピッチ	×	×
SUB E	1A, 45	漢字30ドットピッチ	×	×
SUB P	1A, 50	漢字36ドットピッチ	×	×
SUB 0	1A, 30	「0」字体を「Ø」に変換	×	×
SUB a	1A, 61	インクリボン交換位置への移動	×	×
SUB b	1A, 62	キャリッジの初期化	×	×

4. その他拡張制御コード（STXシーケンス）

シンボル	HEXコード	機　能	TYPE A	TYPE B
STX P	02, 50	書式印字モードの設定	×	×
STX H	02, 48	ヘッドの位置を指定された絶対位置へ移動（n/180インチ）	×	×
STX V	02, 56	用紙を指定された絶対位置へ移動（n/120インチ）	×	×
STX R	02, 52	印字開始位置をダウンロードする	×	×
STX Q	02, 51	用紙吸入時のヘッド位置を指定	×	×
STX T	02, 54	登録された印字開始位置にテスト印字する	×	×
STX I	02, 49	イニシャルリセット	×	×
STX K n	02, 4B	ハガキ印字モード指定	×	×

5. その他拡張制御コード（FS シーケンス）

シンボル	HEX コード	機　能	TYPE A	TYPE B
FS A	1C, 41	FS B によるドットスペースコントロールの解除	×	×
FS B	1C, 42	ドットスペースコントロール	×	×

注　意　○は必須
×はオプション（サポートしなくてもよい。ただし、シーケンスはその機能に予約される）

6.2.2 制御コード

CR　　印字指令　　0DH

印字開始指令コードであり、印字位置は行の先頭に戻ります。縦倍角を印字した場合、印字後II位置へCRします。

I		1パス印字後15/120インチ改行
II		2パス目印字

LF　　改行　　0AH

プリンタバッファ内のデータを印字終了後1行改行します。

順方向改行モードのとき、VFUにボトム位置が設定されていると、ボトム位置からの改行TOF（Top Of Feed）位置となります（ボトム領域の自動フィード）。逆方向改行モードのとき、ボトム位置が設定されていると、TOF位置からの改行はボトム位置となります(ボトム領域の自動フィード)。VT、FF、US、バッファフル印字の自動復改を行うときも同じです。

しかし、ACSF（Auto Cut Sheet Feeder）使用時は、TOF位置を越える改行はTOF位置に停止し、順改行モードになるまでLFを無視します。また、1インチ以上の逆改行はすべて1インチ逆改行になります。

VT　　垂直タブ　　0BH

多行送りコードを意味し、後述のVFUのチャンネル2にセットされているタブ位置まで改行します。

電源投入およびリセット時では、チャンネル2は1インチ毎にセットされているので、VTを受信すると最大6行改行します。また、チャンネル2に何もセットされていないときは、TOF位置まで改行します。逆方向改行モードのときは、ボトム位置まで逆改行します。

FF　　フォームフィード　　0CH

プリンタバッファ内のデータを印字後、VFUのチャンネル1にセットされてい

るTOF位置まで改行します。フロントパネルに改頁スイッチがある場合、そのスイッチと同じ機能をします。

改行中PE (Paper Empty) となったときは、改ページ終了後OFF LINEになります。また、ACSF時は用紙を排出し次の用紙を吸入します。

SO　拡大印字モード指定　0EH

拡大（横2倍）印字指令コードで、どんな印字モード（イメージ印字など）に対しても次のSIを受信するまでは拡大文字になります。

文字の一部が右マージン位置からはみだしたときは、その文字は改行してから次行の先頭に印字されます。

SI　拡大印字モード解除　0FH

拡大（横2倍）印字モードを解除します。

HT　水平タブ　09H

1行中で、設定された水平タブで現在の位置から最も近い位置までキャリッジが移動します。水平タブが設定されていない場合は無視されます。

水平タブは設定されたときの絶対位置として記憶され、設定後印字モードを変更してもその位置は変化しません。

CAN　キャンセル　18H

このコードを受信したときの行の印字データを消去し、受信位置をその行の第1文字目にします。ただし、このコード以前に受信したコントロールコードは消去されません。

また、何らかの印字条件が成立し印字した場合は、受信位置はその行の先頭となり重ね印字をします。

VFU (Vertical Format Unit)

VFUとは、ページ長、垂直タブ位置を設定、実行するもので、タブ位置は独立に

6つの場所の設定が可能です。この6つの独立したタブ位置はそれぞれチャンネル1(CH1)、チャンネル2(CH2)〜チャンネル6(CH6)とよび、US(1FH+n)によってこの実行を指定します。

TOF位置はVFUがセットされたときの受信位置となり、VFUのセットは1インチを単位として行います。

VFUのデータは2バイトで1セットです。また、VFUの設定時には、第1行目と最終行+1行目は必ずTOF(41H)になり、これ以外の指定はエラーになります。

ボトム領域はCH1とCH2で指定します。ボトムの指定をした後のチャンネル設定は無効です。ページ長は最大12インチです。

2度目のCH1以降にチャンネルセットがされるとエラーになるので、VFUデータの設定が終了したら、必ずRSを送出しなければなりません。エラーの発生した時点でこのシーケンスは終了し、VFUを初期値にし、以降は印字データとして受信します。

VFUの設定の詳細は、各プリンタのマニュアルを参照して下さい。

GS　VFU設定の開始　1DH

以後の受信データをVFU設定データとして受信します。

RS　VFU設定の終了　1EH

このコードの受信によりVFU設定のシーケンスを終了します。

US　VFUの実行　1FH+n

プリンタバッファ内の印字データを印字後、VFUでセットされたチャンネルnまで用紙送りを実行します。

次のチャンネルnがセットされていないときは、TOFまで用紙送りを実行します。逆方向改行モードになっていると逆方向に改行します。

US　n行改行　1FH+n

プリンタバッファ内の印字データを印字後、(n-16)行の改行をします。nの範囲は、17≦n≦31で範囲外のときは無効となります。また、ボトム領域にかかると改頁

し、以後の改行は無視します。逆方向改行モードになっていると逆方向に改行します。

EOT　外字登録データの終了　04H

外字登録データを終了します。外字登録数は TYPE A は 85 文字以上、TYPE B は 20 文字以上です。

SOH　グラフィックキャラクターの指定　01H

このコード受信により次の1バイトがグラフィックキャラクタとなります。印字データはグラフィックキャラクタの文字コードに 40H を加えたものとなります。

このデータが 40H～5FH の範囲外のとき SOH コードは無視されます。

6.2.3 拡張制御コード（ESCシーケンス）

ESC N　パイカモードの設定　1BH+4EH

パイカ(10CPI)印字モードになります。高速印字モードのときはHSパイカ、高密度のときはHDパイカになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC H　パイカモードの設定　1BH+48H

ESC Nと同じで、パイカ（10CPI）印字モードになります。高速印字モードのときはHSパイカ、高密度のときはHDパイカになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC Q　コンデンスモードの設定　1BH+51H

コンデンス（18CPI）印字モードになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC E　エリートモードの設定　1BH+45H

エリート(12CPI)印字モードになります。高速印字モードのときはHSエリート、高密度のときはHDエリートになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC P　プロポーショナルモードの設定　1BH+50H

プロポーショナル(P.S)印字モードになります。ANKのキャラクタ幅(大きさ)が各文字により異なり、1行の印字桁数は印字文字により変化しますが、文字と文字の間隔が2ドットスペースと一定となるためバランスのとれた印字になります。

マージン、水平タブなどの設定時には、文字の大きさはパイカ(10CPI)の文字幅として扱われます。また、カタカナ、ひらがな、グラフィックのときは10CPIとなり、スペース（20H）は（5+2）ドットの大きさになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC K　漢字モード（横印字）　1BH+4BH

漢字横書き印字モードになります。以後、テキストデータは漢字コードとして受信されます。漢字コードは2バイトで指定し、それぞれ20H〜7FHの範囲でなければなりません。また、第1バイトが20H未満のときはコントロールコードとなり、次の1バイトが第1バイトになります。そして、第2バイトが20H未満または80H以上のときはデータを無視します。半角文字も漢字印字と同様に行います。半角コードの範囲は0020H〜00FFHになります。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC t　漢字モード（縦印字）の設定　1BH+74H

漢字縦書き印字モードになります。半角文字の縦書きは、ESC h 1で縦書き指定をしないと横書きで印字します。

他の印字指定コードを受信するまでこのモードは解除されません。

ESC n　nドットスペース　1BH+n

プロポーショナルモードおよび漢字モード時のみ有効となり、その他の印字モードでは無効になります。このコードを受信すると、現在の横ドットピッチのnドット/120インチ、nドット/160インチ、nドット/180インチ（nは1〜8）のいずれかのスペースが挿入されます。nが0のときはコードを無視し、9以上のときはESCシーケンスと見なします。

この指定は文字単位であり、コード受信後すべての文字間に挿入されることはありません。拡大印字モードの場合はそれぞれが2倍のドットスペースになります。

ESC S　8ドットビットイメージモードの設定　1BH+53H+n3n2n1n0+イメージデータ

縦8ドットビットイメージモードになります。4バイトのデータ(n3n2n1n0)により、ビットイメージのバイト数を表します。

n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。なお、指定バイト数のデータを受信すると、自動的にドット対応グラフィック以前に指定されているキャラクタモードに復帰します。

ESC I　16ドットビットイメージモードの設定　1BH+49H+n3n2n1n0+イメージデータ

縦16ドットビットイメージモードになります。4バイトのデータ（n3n2n1n0）により、ビットイメージのバイト数を表します。ただし、縦16ドットビットイメージモードでは、2バイトで1ドット列となるので、(n3n2n1n0)×2がデータバイト数です。

n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。なお、指定ドット列数のデータを受信すると、自動的にそれ以前に指定されているキャラクタモードに復帰します。

ESC J　24ドットビットイメージモードの設定　1BH+4AH+n3n2n1n0+イメージデータ

縦24ドットビットイメージモードになります。4バイトのデータ（n3n2n1n0）により、ビットイメージのバイト数を表します。ただし、縦24ドットビットイメージモードでは、3バイトで1ドット列となるので、(n3n2n1n0)×3がデータバイト数です。

n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。なお、指定ドット列数のデータを受信すると自動的にそれ以前に指定されているキャラクターモードに復帰します。

注　意　ビットイメージモードのn3n2n1n0は10進4桁の数字で、各nは30H≦n≦39Hの範囲でなければなりません。

ESC V　8ドットビットイメージリピート　1BH+56H+n3n2n1n0+D1

D1という8ドットのデータをn3n2n1n0で示される回数だけ繰り返して印字します。n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。ヘッドピンに対するドット対応は、ESC Sと同じです。

ESC W　16ドットビットイメージリピート　1BH+57H+n3n2n1n0+D1D2

D1D2という16ドットのデータをn3n2n1n0で示される回数だけ繰り返して印字します。n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。

ヘッドピンに対するドット対応は、ESC Iと同じです。

ESC U　24ドットビットイメージリピート　1BH+56H+n3n2n1n0+D1D2D3

D1D2D3という24ドットのデータをn3n2n1n0で示される回数だけ繰り返して印字します。n3n2n1n0は必ず4桁の10進数で、その範囲は0001〜9999です。

ヘッドピンに対するドット対応は、ESC Jと同じです。

ESC F　ドットアドレッシング　1BH+46H+n3n2n1n0

一時的なドットスペースのタブ（ドットアドレッシング）を行います。4バイトのデータ（n3n2n1n0）により、ドットアドレッシングの位置を指定します。ドットアドレッシングはレフトマージン（LM）位置を0とした相対アドレスで行います。現在の受信位置よりも小さなドットアドレッシングは無効になります。

ESC A　1/6インチ改行　1BH+41H

改行幅を1/6インチにします。電源投入時は、この改行幅となっています。

他の改行指定コードを受信するまでこの指令は解除されません。

ESC B　1/8インチ改行　1BH+42H

改行幅を1/8インチにします。

他の改行指定コードを受信するまでこのモードは解除されません。ただし、画面コピーモードにおいて、SUB A、SUB Bでは24/180インチ、SUB C、SUB D、ESC Dでは16/180インチ改行になります。

ESC T　n/120インチ改行またはn/180インチ改行　1BH+54H+n1n0

改行幅を指定された基本改行ピッチの1/120インチまたは1/180インチ単位で任意に設定します。n1n0の範囲は0≦n1n0≦99です。ESC D、ESC Mモードでは1/180インチに固定されます。1/120インチ、1/180インチの切り換えは、SUB G、SUB Fで行います。

他の改行指定コードを受信するまでこのモードは解除されません。

ESC (　水平タブセット　1BH+28H+n2n1n0+2CH+・・・2EH

任意の所に水平タブ（HT）をセットします。HTは最大36ケ所まで指定できます。指定後、HT(09H)を送信することにより、キャリッジはHT設定位置まで進みます。

HTは設定時の印字モード(パイカ、エリート、コンデンス、プロポーショナル、漢字）の文字幅で設定されます。設定後、印字モードを変更し、文字幅が変更しても設定位置は変化しません。設定位置は、レフトマージンを基準に、左端の文字を1とし3桁の10進数で表します。また、2CHで継続され、設定終了コードとして2EHで閉じて設定されます。

ただし、第1桁目に設定することはできません。

ESC)　水平タブ部分クリア　1BH+29H+n2n1n0+2CH+～2EH

設定された水平タブ(HT)を部分的にクリアします。HTを設定した印字モードで解除して下さい。設定時の印字モードと異なる印字モードにて解除した場合は解除できないことがあります。解除位置は、レフトマージンを基準に、左端の文字を1とし3桁の10進数で表します。

また、2CHで継続され、解除終了コードとして2EHで閉じて解除されます。

ESC 2　水平タブオールクルア　1BH+32H

設定された水平タブ（HT）を全て解除します。以後、HT（09H）を受信してもHT動作を行いません。

ESC X　アンダーライン開始　1BH+58H

アンダーライン開始指令コードです。このコード受信後は、全ての印字モードに対してアンダーラインを印字します。アンダーライン解除指令コードを受信するまで、このモードは解除されません。アンダーラインの位置は、印字後2/220インチ改行し、ヘッド23、24ピンを使用して印字します。水平タブによる移動に対してはアンダーラインは印字されません。ESC Fによる受信位置の移動に対してはアンダーラインが付加されます。

アンダーラインのドットピッチは、HD印字中は1/180インチ、HS印字中は1/90インチになります。

ESC Y　アンダーライン解除　1BH+59H

このコード受信後、アンダーライン印字は解除されます。

ESC L　レフトマージンの設定　1BH+4CH+n2n1n0

レフトマージンを設定します。マージン幅は、現在の印字モードの文字幅単位で設定されます。3ケタの10進数（n2n1n0）でマージン幅を指定します。

なお、プロポーショナルモードのときは、パイカと同様の文字幅になります。他のレフトマージン設定が行われるまで印字開始位置は変化しません。電源投入時にはレフトマージンは「000」になります。

レフトマージンの設定はホームポジションを基準に設定され、設定後の印字領域が72ドット/180インチ未満のときと、ライトマージンを越える場合は無視されます。

ESC /　ライトマージンの設定　1BH+2FH+n2n1n0

ライトマージンを設定します。マージン幅は、現在の印字モードの文字幅単位で設定されます。3ケタの10進数（n2n1n0）でマージン幅を表します。

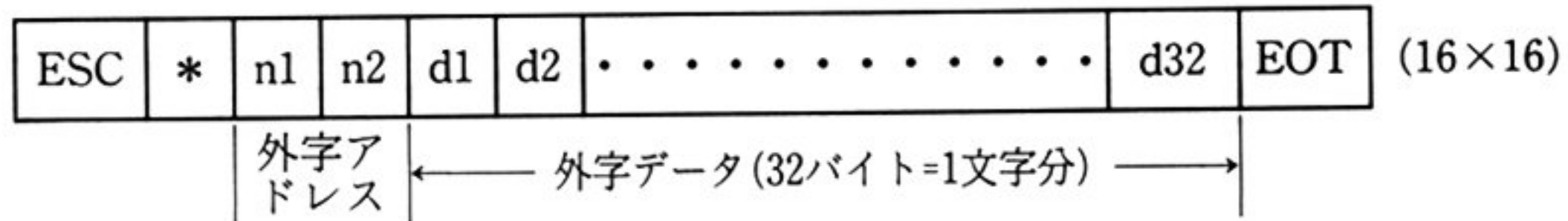
なお、プロポーショナルモードのときは、パイカと同様の文字幅になります。他のライトマージン設定が行われるまで設定されたライトマージンは解除されません。

ライトマージンの設定はホームポジションを基準に設定され、設定後の印字領域が72ドット/180インチ未満の場合に、レフトマージンを解除すれば72ドット以上の印字領域がとれるときは、レフトマージンが解除されます。
また、レフトマージンを解除しても72ドット以上の印字領域が確保できないときは、この指定は無視されます。

指定が印字の最大桁数を越えているとき、または等しいときはこの指定は解除されます。

ESC *　外字の登録（16×16）　1BH+2AH+n1n0+d1d2+～+04H

16×16ドットの外字を登録します。外字機能を使うことにより、自由な文字を印字することができます。

外字文字のドットは縦、横16×16ドットで構成されるドットマトリックスを24×24ドットマトリックスに変換して登録します。外字ロードの書式は次のようになります。

ESC	*	n1	n2	d1	d2	・・・・・・・・・・・・・	d32	EOT

(16×16)

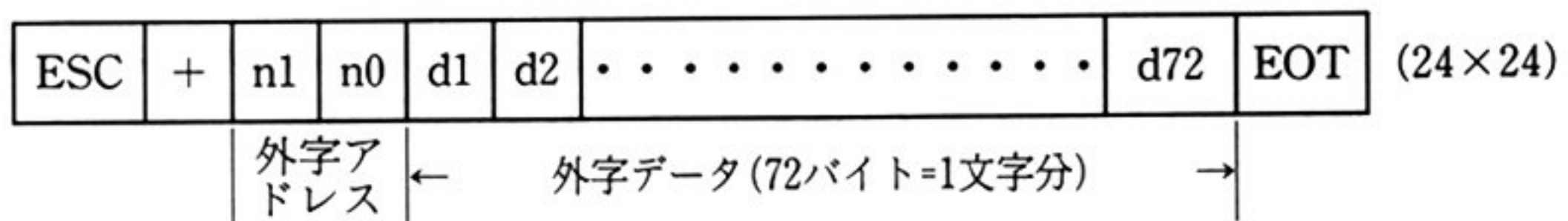
外字アドレス ←―― 外字データ(32バイト=1文字分) ――→

n1n0により外字を定義する漢字コードを指定します。n1n0の範囲はそれぞれ20H～7FHの範囲で指定でき、外字が定義されると、その漢字コードに対しては定義した外字を印字します。定義が解除されると漢字ROM中のそのコードに対応したデータを印字します。

2921H～297EH、2A21H～2A7EH、2B21H～2B7EHのエリアは半角文字エリアですので、24×24ドットマトリックスに変換された1～12ドット列までのデータしか印字されません。ただし、登録する場合には、32バイトのデータが必要です。16ビットドット対応グラフィックと同じ要領でデータを作成して下さい。

EOT (04H) は外字のロードを終了するコードですが、なくても自動的に外字登録シーケンスを終了します。

ESC ＋　外字の登録（24×24）　1BH+2BH+n1n0+d1d2+～+04H

ドット構成が24×24の外字の登録をします。n1n0の2バイトが外字登録アドレスです。そして、それに続く72バイトが登録データです。外字ロードの書式は次のようになります。

ESC	+	n1	n0	d1	d2	・・・・・・・・・・・・・	d72	EOT

(24×24)

外字アドレス ← 外字データ(72バイト=1文字分) →

n1n0により外字を定義する漢字コードを指定します。n1n0の範囲はそれぞれ20H～7FHの範囲で指定でき、外字が定義されると、その漢字コードに対しては定義した外字を印字します。定義が解除されると漢字ROM中のそのコードに対応したデータを印字します。

2921H～297EH、2A21H～2A7EH、2B21H～2B7EH（TYPEBは7621H～7633H)のエリアは半角文字エリアですので24×24ドットの1～12ドット列までのデータしか印字されません。ただし、登録する場合には、72バイトのデータが必

要です。24 ビットドット対応グラフィックと同じ要領でデータを作成して下さい。

EOT (04H) は外字のロードを終了するコードですが、なくても自動的に外字登録シーケンスを終了します。

ESC D　1/90 インチイメージ印字モード指定(コピーモード)　1BH+44H

ドット列印字モードにおける基本ドット列ピッチを 1440 ドット / 8 インチに設定します。ただし、ESC S (8 ドットビットイメージモード) の場合は基本ドット列ピッチは 720 ドット / 8 インチになり、MSX の画面を印字する際に、縦と横の比率が 1：1 に修正されます。

このモードは他の基本ドット列ピッチ指定 (SUB A、SUB B、SUB D) を受けるまで解除されません。

ESC M　1/180 インチイメージ印字モード指定(コピーモード解除)　1BH+4DH

ドット列印字モードにおける基本ドット列ピッチを 1440 ドット / 8 インチに設定します。このモードは他の基本ドット列ピッチ指令 (SUB A、SUB B、SUB D) を受けるまで解除されません。

ESC R　キャラクタリピート　1BH+52H+n2n1n0+D

指定されたキャラクタを指定された数だけ印字します。3 バイトのデータ (n2n1n0) でリピート数を指定し、次の 1 バイト D は印字したいキャラクタを指定します。n2n1n0 は必ず 3 桁の 10 進数で、その範囲は 001～999 です。

D がコントロールコードだったときは、以前の設定は無効となりそのコードが実行されます。この命令は、漢字印字モードのときは無効です。

ESC !　強調印字モードの指定　1BH+21H

強調印字モードに指定されます。強調印字は 1 度印字し、またその上を再度印字することなどにより行います。

この指定は、全ての印字モードに対し有効で、解除指令コードが受信されるまで強調印字が続けられます (イメージ印字に対しても有効)。

ESC " 強調印字モード解除 1BH+22H

強調印字モード解除モードを解除します。

ESC > 片方向印字モード指定 1BH+3EH

全ての印字モードで、印字方向が左から右の片方向印字になります。このモードは片方向印字モード解除コードを受信するまで続きます。

ESC] 片方向印字モード解除 1BH+5DH

片方向印字モードを解除し、双方向最短距離印字にします。

ESC f 順方向改行モード 1BH+66H

改行方向を順方向とします。全ての改行動作(VFU、改行スイッチ、改頁スイッチ)に対して有効になります。電源投入時はこのモードになっています。

逆方向指定コードを受信するまでこのモードは解除されません。

ESC r 逆方向改行モード 1BH+72H

改行方向を逆方向とします。改行スイッチ、改頁スイッチに対しては無効になります。

順方向改行指定コードを受信するまでこのモードは解除されません。

ESC a 用紙の排出吸入 1BH+61H

このコードはACSF時のみ有効で、用紙の排出吸入動作を行います(FFコードと同じ)。

ESC b 用紙の排出 1BH+62H

このコードはACSF時のみ有効で、用紙の排出動作を行います。バッファ内に印字データがあるときは、データを印字後、排出を行います。ただし、用紙がないときは吸入後、印字して排出します。

ESC C S スーパースクリプトモード 1BH+43H+53H

スーパースクリプトモード印字を行います。文字幅は20文字/インチです。
他の印字指定コードを受信するまでこのコードは解除されません。

ESC C s サブスクリプトモード 1BH+43H+73H

サブスクリプトモード印字を行います。文字幅は20文字/インチです。
他の印字指定コードを受信するまでこのコードは解除されません。

ESC s 1 スーパースクリプトモード 1BH+73H+31H

スーパースクリプトモード印字を行います。文字幅は20文字/インチです。
他の印字指定コードを受信するまでこのコードは解除されません。

ESC s 2 サブスクリプトモード 1BH+73H+32H

サブスクリプトモード印字を行います。文字幅は20文字/インチです。
他の印字指定コードを受信するまでこのコードは解除されません。

ESC h 1 半角文字縦印字指定 1BH+68H+31H

半角文字の縦印字を行います。この指定は漢字モード（縦印字）時においてのみ有効です。

ESC h 0 半角文字縦印字解除 1BH+68H+30H

半角文字の縦印字を解除し、半角文字を横印字にします。

ESC q 半角組文字縦印字指定 1BH+71H

漢字モード（縦印字）時でさらに半角縦印字モード中、このシーケンスに続く半角2文字をペアで縦印字します。この指定は半角2文字を印字後、自動的に解除されます。また、半角文字が1文字の場合はただちに解除されます。

ESC e n1 n2　文字拡大率指定　1BH+65H+n1+n2

文字拡大率指定を表します。n1、n2は選択数値パラメータで文字の大きさを指定します。単位はn倍で、n1は縦方向の、n2は横方向の倍率を指定します。n1、n2とも1（31H）または2（32H）が指定可能です。

ESC d n　ドラフトモードの切り換え　1BH+64H+n

ドラフトモードの指定、解除を選択します。nは数値パラメータでパラメータによりドラフト印字モードと通常の印字モードを選択することができます。ドラフト印字モードは印字品質を落とすことにより、高速に印字することができます。

nのLSB（Least Significant Bit）により以下のモードが選択できます。

LSB=1	31H	高速印字文字
LSB=0	30H	高密度印字文字（通常印字モード）

ソフトウェアがこのコマンドを使用する場合、このコマンドをサポートしていないプリンタに余分な文字を印字させない為に、以下のコマンドを使用するようにしてください。

ドラフト印字	CR	ESC	d	CR(0DH)
通常印字に復帰	CR	ESC	d	CAN(18H)

ESC c 1　ソフトウエアリセット　1BH+63H+31H

プリンタを初期状態にします。

ESC C J　JISモードの指定　1BH+43H+4AH

1983年度版JISコードのモード指定を行います。JISモードでは、コード体系はJISに準拠します。

ESC C j　JISモードの解除　1BH+43H+6AH

MSXのモード指定を行います。MSXモードでは、コード体系はMSX本体の取扱説明書などの漢字コード表にしたがって下さい。

MSX漢字コード表は1979年版JISのスーパーセットになっています。1983年版JISとは1区と2区、8区が異なります。

ESC @　紙送りのバックラッシュ除去　1BH+40H

紙送りのバックラッシュを除去します。

ESC x　漢字フォントの転送　1BH+78H

漢字フォントを転送します（割り込みなし）。

ESC y　漢字フォントの転送　1BH+78H

漢字フォントを転送します（割り込み有り）。

ESC C a　A4 用紙の印字　1BH+43H+61H

A4 用紙への印字モードになります。

ESC C b　B4 用紙の印字　1BH+43H+62H

B4 用紙への印字モードになります。

ESC C t　葉書縦挿入時の紙送り初期化　1BH+43H+74H

葉書縦挿入時の紙送りを初期化します。

ESC C y　葉書横挿入時の紙送り初期化　1BH+43H+79H

葉書横挿入時の紙送りを初期化します。

ESC C T　テスト印字モード　1BH+43H+54H

テスト印字モードになります。

ESC p `　漢字モードで英数記号の印字を標準書体に設定　1BH+70H+60H

漢字モードで英数記号の印字を標準書体（明朝体）に設定します。

ESC p a 漢字モードで英数記号の印字をポエム体に設定 1BH+70H+61H

漢字モードで英数記号の印字をポエム体に設定します。

ESC p b 漢字モードで英数記号の印字をデコ体に設定 1BH+70H+62H

漢字モードで英数記号の印字をデコ体に設定します。

ESC p c〜ESC p _ 予約 1BH+70H+63H〜7EH

漢字モードでの英数記号の印字をマルチフォント化するために、「ESC p b」（HEX コードの 1B+70+63）から「ESC p _」（HEX コードの 1B+70+7E）までは予約します。

ESC p @ 漢字モードで平、片仮名の印字を標準書体に設定 1BH+70H+40H

漢字モードで平、片仮名の印字を標準書体（明朝体）に設定します。

ESC p A 漢字モードで平、片仮名の印字をポエム体に設定 1BH+70H+41H

漢字モードで平、片仮名の印字をポエム体に設定します。

ESC p B 漢字モードで平、片仮名の印字をかしこ体に設定 1BH+70H+42H

漢字モードで平、片仮名の印字をかしこ体に設定します。

ESC p C〜ESC p _ 予約 1BH+70H+43H〜5FH

漢字モードでの平、片仮名の印字をマルチフォント化するために、「ESC p B」（HEX コードの 1B+70+43）から「ESC p _」（HEX コードの 1B+70+5F）までは予約します。

ESC p Z 漢字モードでの印字をすべて標準書体に設定 1BH+70H+5AH

漢字モードでの印字をすべて標準書体（明朝体）に設定します。

ESC p 0 文字飾り印字の解除 1BH+70H+30H

文字飾り印字を解除します。

ESC p 1 白抜き印字の設定 1BH+70H+31H

すべての印字を白抜きにします。

ESC p 2 斜体印字の設定 1BH+70H+32H

すべての印字を斜体にします。

ESC p 3 太字印字の設定 1BH+70H+33H

すべての印字を太字にします。太字は基本フォントパターンと1ドットシフトさせたパターンのORを取ります。そのため、強調印字とは異なります。

ESC p 4〜ESC p ? 予約 1BH+70H+34H〜3FH

文字飾り印字のために、「ESC p 4」（HEXコードの1B+70+34）から「ESC p ?」（HEXコードの1B+70+3F）までは予約します。

ESC p !〜ESC p / 予約 1BH+70H+21H〜2FH

文字飾り印字のために、「ESC p !」（HEXコードの1B+70+21）から「ESC p /」（HEXコードの1B+70+2F）までは予約します。

6.2.4 拡張制御コード（SUBシーケンス）

SUB A　960ドット/8インチ基本ドット列ピッチ　1AH+41H

ビットイメージモードにおける基本ドット列ピッチを960ドット/8インチに設定します。

このモードは他の基本ドット列ピッチ指令SUB B、SUB C、SUB Dを受けるまで解除されません。

SUB B　1280ドット/8インチ基本ドット列ピッチ　1AH+42H

ビットイメージモードにおける基本ドット列ピッチを1280ドット/8インチに設定します。

このモードは他の基本ドット列ピッチ指令SUB A、SUB C、SUB Dを受けるまで解除されません。

SUB C　1440ドット/8インチ基本ドット列ピッチ　1AH+43H

ビットイメージモードにおける基本ドット列ピッチを1440ドット/8インチに設定します。

このモードは他の基本ドット列ピッチ指令SUB A、SUB B、SUB Dを受けるまで解除されません。

SUB D　720ドット/8インチ基本ドット列ピッチ　1AH+43H

8ドット対応グラフィックの基本ドット列ピッチを720ドット/8インチとし、他の印字モードにおける基本ドット列ピッチを1440/8インチに設定します。

このモードは他の基本ドット列ピッチ指令ESC M、ESC D、SUB A、SUB B、SUB Cを受けるまで解除されません。

SUB F　120インチ改行ピッチモード　1AH+46H

このコードを受信すると、以後ESC Tにおける改行ピッチコマンドの基本改行

ピッチが120インチになります。

このモードはSUB G (180インチ改行ピッチ) コマンドを受けるまで解除されません。

SUB G　180インチ改行ピッチモード　1AH+47H

このコードを受信すると、以後ESC Tにおける改行ピッチコマンドの基本改行ピッチが180インチになります。

このモードはSUB F (120インチ改行ピッチ) コマンドを受けるまで解除されません。

SUB 1　外字登録部分クリア　1AH+31H+n1n0

2バイトの漢字コード (n1n0) 上に登録された外字をクリアします。

外字登録がクリアされた場合、そのコードが内部に実装されている漢字ROMのデータエリア内のコードである場合には、漢字ROMの内容が選ばれます。

n1n0によって指定された外字が登録されていないときは、このコードは無視されます。

SUB 2　外字登録オールクリア　1AH+32H

登録されている外字データを全てクリアします。

SUB U　スーパースクリプトモード　1AH+55H

スーパースクリプトモード印字を行います。文字幅は20文字/インチです。

他の印字指定コードを受信するまでこのコードは解除されません。

SUB L　サブスクリプトモード　1AH+4CH

サブスクリプトモード印字を行います。文字幅は20文字/インチです。

他の印字指定コードを受信するまでこのコードは解除されません。

SUB V　縦拡大印字モード指定　1AH+56H

全ての印字モードで、文字を縦方向に2倍に拡大して印字します。このモードは、上側のデータを印字して、15/120インチ改行後下半分を印字します。

この改行に対する補正は行わないので、ユーザーはプログラムによってその補正をしなければなりません。

縦拡大印字解除指令コードを受信するまでこのコードは解除されません。

SUB W　縦拡大印字モード解除　1AH+57H

指定された縦拡大印字モードを解除します。

SUB Q　漢字ドットピッチ 24　1AH+51H

漢字1文字を横24ドットピッチモードにします。文字幅は7.5文字/インチとなり、1行60文字の印字ができます。半角文字の場合、12ドットピッチになります。

他の漢字ドットピッチモード（SUB N、SUB E、SUB P）が指定されるまでこのモードは解除されません。

SUB N　漢字ドットピッチ 27　1AH+4EH

漢字1文字を横27ドットピッチモードにします。文字幅は6.66文字/インチとなり、1行53文字の印字ができます。半角文字の場合、13または14ドットピッチになります。

他の漢字ドットヒッチモード（SUB Q、SUB E、SUB P）が指定されるまでこのモードは解除されません。

SUB E　漢字ドットピッチ 30　1AH+45H

漢字1文字を横30ドットピッチモードにします。文字幅は6.0文字/インチとなり、1行48文字の印字ができます。半角文字の場合、15ドットピッチになります。

他の漢字ドットピッチモード（SUB Q、SUB N、SUB P）が指定されるまでこのモードは解除されません。

SUB P　漢字ドットピッチ 36　1AH+50H

漢字1文字を横36ドットピッチモードにします。文字幅は5.0文字/インチとなり、1行40文字の印字ができます。半角文字の場合、18ドットピッチになります。

他の漢字ドットヒッチモード（SUB Q、SUB N、SUB E）が指定されるまでこのモードは解除されません。

SUB 0　「O」字体の切り換え　1AH+30H

「0」字体を「0」から「Ø」に切り換えます。一度切り換えると電源を切るまで換わりません。

SUB a　インクリボン交換位置への移動　1AH+61H

インクリボンの交換位置へ移動します。

SUB b　キャリッジの初期化　1AH+62H

キャリッジの初期化を行ないます。

6.2.5 拡張制御コード（STXシーケンス）

STX P　書式印字モード設定　02H+50H

書式印字モードになります。この命令が設定されると、プリンタのメモリ上の各種の設定は、外字ロードを除いてクリアされます。また、データエリアとして外字ロードエリアの一部を使用するため、5文字の外字データが消滅することがあります。

このコマンドは書式印字モード時は無視されます。

STX H　ヘッドの水平位置移動　02H+48H+n1+n2

ヘッドの位置を印字の左端（ホームポジション）より、n1×256+n2ドット目に移動させます。パラメータの範囲はn1×256+n2≦1439（単位は1/180インチ）で、範囲外のときは無視されます。

STX V　ヘッド垂直位置移動　02H+56H+n1+n2

ヘッドをTOFより、n1×256+n2+63の位置へ移動させます。パラメータの範囲は、15≦n1×256+n2≦1319（単位は1/120インチ）で、パラメータの範囲が15より小さい場合は15に設定され、1320以上のときはコマンドを無視します。

STX R　書式データのダウンロード　02H+52H+l+Si+ni1+ni2+mi1+mi2

1個の書式データをダウンロードします。1ケ所の書式データは、下記のように5バイトで構成されます。

l	書式データの個数
Si	下位4ビットで、ドットスペースを表し、上位ビットは無視
ni1、ni2	登録の垂直位置（単位は1/120インチ）
mi1、mi2	登録の水平位置（単位は1/180インチ）

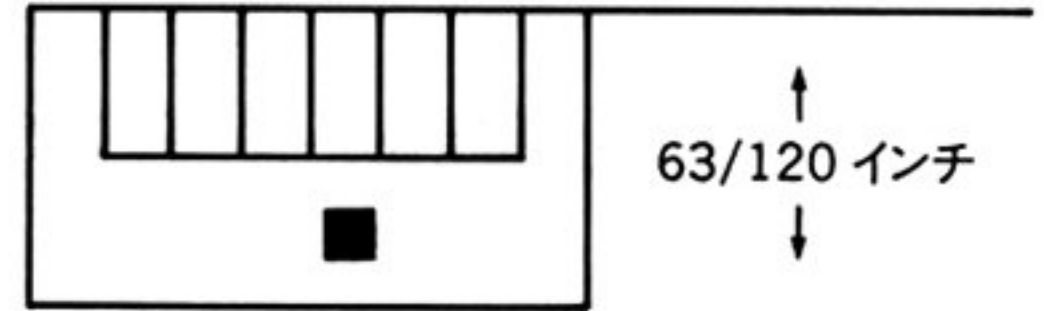
(i = 1、2、・・・・・・、l−1、l)

以前に登録のあった書式データはクリアされます。

パラメータ範囲は、

1≦l≦60

63≦ni1×256＋ni2≦1319

0≦mi1×256＋mi2≦1439

で、lがこの範囲外のとき、コマンドはその時点で終了し、無視されます。位置が指定範囲外のときは、その最大値、最小値にセットします。

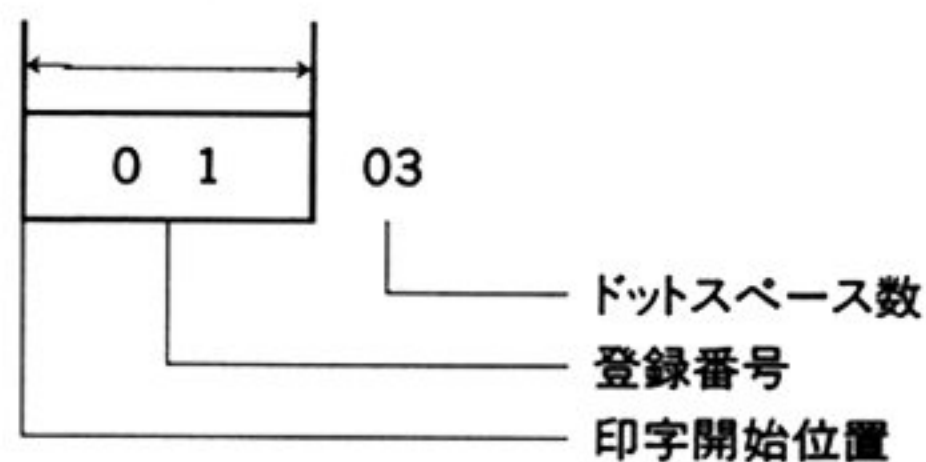
STX Q　センタリング位置の指定　02H+51H+n1+n2

キャリッジを印字の左端より（n1×256＋n2）ドットの位置（センタリング位置）へ移動します。そして、以後の用紙吸入およびOFF LINE時にキャリッジを設定されたセンタリング位置へ移動します。

パラメータ範囲は、0≦n1×256＋n2≦1439（単位は1/180インチ）であり、範囲外の指定は無視されます。

STX T　テスト印字指定　02H+54H

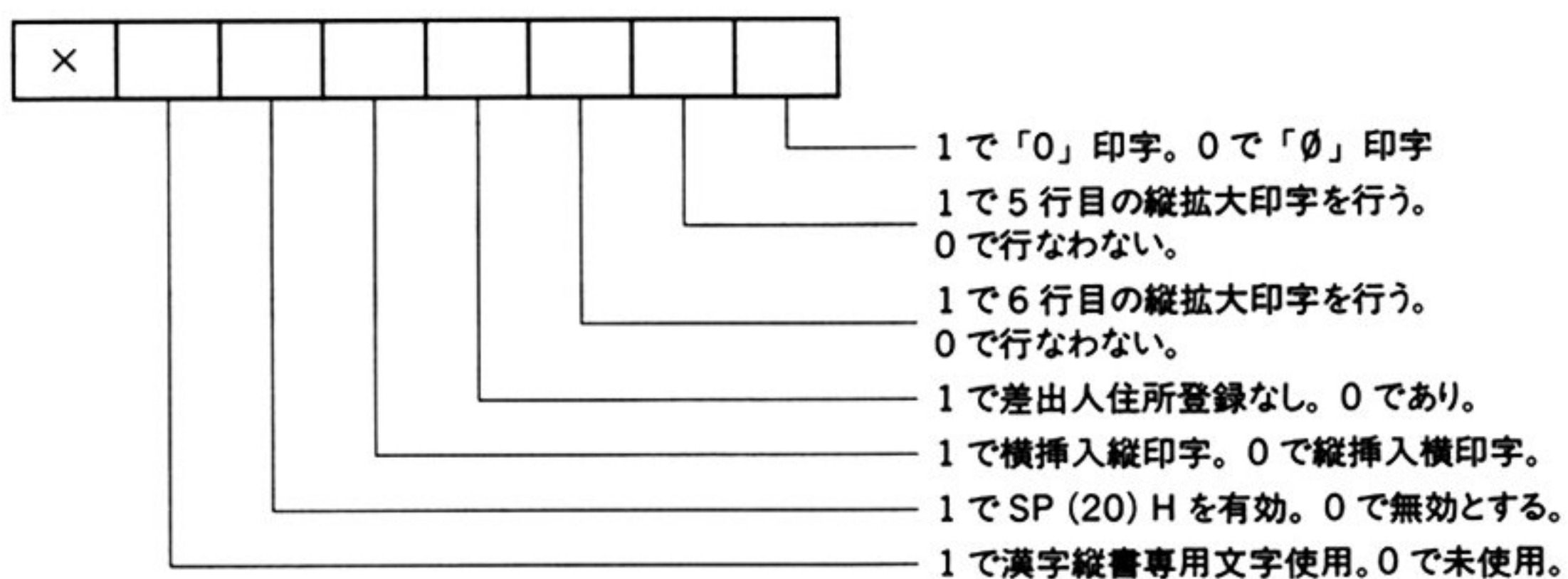
すでにダウンロードされている書式データのテスト印字を行います。テスト印字とは、書式データ登録順に従い、登録番号を印字開始位置に印字します。また、登録番号の後にドットスペース数を印字します。

ただし、印字登録がされていないときはこのコマンドを無視します。

STX K n　葉書印字指定　02H+4BH+n

葉書印字指定を行います。nは1バイトのコードで、各ビットはディップスイッチの各設定に対応します。このコマンドによりバッファ内のデータをクリアし、葉書モードとしてリセット処理を行います。

ここで、ディップスイッチのデータnのMSB～LSBは下図のようになります。

×							

- 1で「0」印字。0で「Ø」印字
- 1で5行目の縦拡大印字を行う。0で行なわない。
- 1で6行目の縦拡大印字を行う。0で行なわない。
- 1で差出人住所登録なし。0であり。
- 1で横挿入縦印字。0で縦挿入横印字。
- 1でSP (20) Hを有効。0で無効とする。
- 1で漢字縦書専用文字使用。0で未使用。

×は未定義で、マスクして無視される。

このコマンドにより一部の外字がクリアされることがあります（5文字分）。

STX I　イニシャルリセット　02H+49H

バッファ内の全印字データおよび全ての設定を初期状態に戻します。

このコマンドにより電源再投入と同じく、書式モード、はがきモード、減音モードはクリアされ標準モードになります。

6.2.6 拡張制御コード（FSシーケンス）

FS B　ドットスペースコントロール　1CH+42H

漢字印字における文字ピッチを32ドットまたは36ドットにします。このモードは漢字文字ピッチ指定（SUBコマンド参照）よりも優先します。受信時のモードにより下記のようになります。

■ハードコピーモード

	前スペース	中ドット	後ろペース
SUB A モード	6	24	6
SUB B モード	6	24	6
SUB C モード	3	24	5
SUB D モード	3	24	5

■ネイティブモード

	前スペース	中ドット	後ろペース
SUB A モード	6	24	6
SUB B モード	6	24	6
SUB C モード	6	24	6
SUB D モード	3	24	5

この指定は他の漢字文字ピッチ指定または、FS Aを受信すると解除されます。

FS A　ドットスペースコントロール解除　1CH+41H

FS Bによるドットスペース指定を解除します。

6.2.7 印字モードに関する注意 (ESC N、ESC H、ESC E、ESC Q、ESC P)

以下にあげるESCシーケンスがハードウェアの制約などにより、やむをえずサポートできない場合、次の点に注意してください。

ESC N、ESC H、ESC E、ESC Q、ESC P

これらのESCシーケンスは印字モードの選択だけでなく、漢字モードの解除にも使用される可能性があります。もし、やむをえずサポートできない場合でも、漢字モードの解除は必ず行って下さい。

6.2.8 イメージ印字

SUBモード	ESCモード	ドットピッチ	使用ドット
SUB A モード	ESC F (ドットアドレッシング)	960ドット/8インチ	24ドット
	ESC V (8ビットドット列リピート)	960ドット/8インチ	24ドット
	ESC S (8ビットドット対応グラフィック)	960ドット/8インチ	24ドット
	ESC W (16ビットドット列リピート)	960ドット/8インチ	24ドット
	ESC I (16ビットドット対応グラフィック)	960ドット/8インチ	24ドット
	ESC U (24ビットドット列リピート)	960ドット/8インチ	24ドット
	ESC J (24ビットドット対応グラフィック)	960ドット/8インチ	24ドット

SUB モード	ESC モード	ドットピッチ	使用ドット
	ESC F (ドットアドレッシング)	1280 ドット/8 インチ	16 ドット
	ESC V (8 ビットドット列リピート)	1280 ドット/8 インチ	16 ドット
	ESC S (8 ビットドット対応グラフィック)	1280 ドット/8 インチ	16 ドット
SUB B モード	ESC W (16 ビットドット列リピート)	1280 ドット/8 インチ	16 ドット
	ESC I (16 ビットドット対応グラフィック)	1280 ドット/8 インチ	16 ドット
	ESC U (24 ビットドット列リピート)	1280 ドット/8 インチ	24 ドット
	ESC J (24 ビットドット対応グラフィック)	1280 ドット/8 インチ	24 ドット

SUB モード	ESC モード	ドットピッチ	使用ドット
	ESC F (ドットアドレッシング)	1440 ドット/8 インチ	16 ドット
	ESC V (8 ビットドット列リピート)	1440 ドット/8 インチ	16 ドット
	ESC S (8 ビットドット対応グラフィック)	1440 ドット/8 インチ	16 ドット
SUB C モード	ESC W (16 ビットドット列リピート)	1440 ドット/8 インチ	16 ドット
	ESC I (16 ビットドット対応グラフィック)	1440 ドット/8 インチ	16 ドット
	ESC U (24 ビットドット列リピート)	1440 ドット/8 インチ	24 ドット
	ESC J (24 ビットドット対応グラフィック)	1440 ドット/8 インチ	24 ドット

SUB モード	ESC モード	ドットピッチ	使用ドット
	ESC F (ドットアドレッシング)	720 ドット/8インチ	16 ドット
	ESC V (8 ビットドット列リピート)	720 ドット/8インチ	16 ドット
	ESC S (8 ビットドット対応グラフィック)	720 ドット/8インチ	16 ドット
SUB D モード	ESC W (16 ビットドット列リピート)	1440 ドット/8インチ	16 ドット
	ESC I (16 ビットドット対応グラフィック)	1440 ドット/8インチ	16 ドット
	ESC U (24 ビットドット列リピート)	1440 ドット/8インチ	24 ドット
	ESC J (24 ビットドット対応グラフィック)	1440 ドット/8インチ	24 ドット

6.3 グラフィック印字

6.3.1 グラフィック印字に関する必須条件と推奨条件

8ドットグラフィックス/16ドットグラフィックスの印字に関しては、24ドットプリンタで完全にサポートすることがむずかしいため、ハードウェアに応じて可能なかぎりサポートするものとします。ただし、その際、必須条件と推奨条件を以下のようにします。

1. グラフィックス印字に関する必須条件

- 上下の行と重なったり離れたりしない。
- 最大印字ドット数が保証される。
- 1：1または8ドットプリンタに近い縦横比で印字可能。

2. グラフィック印字に関する推奨条件

- 8ドットプリンタまたは16ドットプリンタとなるべく同じように印字する。

必須条件とは、リスト7.3やリスト7.4のようなプログラムが正常に印字できることを保証するためのものです。縦横比が1:1であるという条件は、画面コピーのようなプログラムが正常に動作するために必要です。最大印字ドット数が保証されるという条件は、8ドットプリンタ用のワードプロセッサの印字に使用する場合などに必要になります。

推奨条件は、リスト7.5のようなプログラムでも、できれば8ドットプリンタと同じように印字された方が好ましいことを意味します。

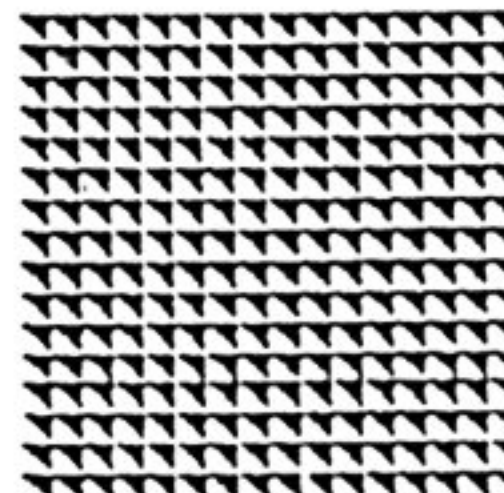
ここに示したプログラムのような場合以外にも、プリンタ用紙に、すでに8ドットプリンタに合わせてフォーマットが印刷されているような場合、この条件が必要になります。

リスト7.3 印字チェックプログラム1

```
100 'TEST PROGRAM NO.1
110 LPRINT CHR$(27);"T16"
120 LPRINT CHR$(27);"N"
130 FOR N = 1 TO 8
140    LPRINT CHR$(27);"S0064";
150    FOR M = 1 TO 64
160       LPRINT CHR$(2^(M MOD 8) - 1);
170    NEXT M
180    LPRINT
```

```
190 NEXT N
200 END
```

リスト 7.4 印字チェックプログラム 2

```
100 'TEST PROGRAM NO.2
110 LPRINT CHR$(27);"T16"
120 LPRINT CHR$(27);"N"
130 FOR N = 1 TO 8
140   LPRINT CHR$(27);"I0128";
150   FOR M = 1 TO 255
160     LPRINT CHR$(2^(M/2 MOD 8)-1)
170   NEXT M
180   LPRINT
190 NEXT N
200 END
```

リスト 7.5 印字チェックプログラム 3

```
100 'TEST PROGRAM NO.3
110 LPRINT CHR$(27);"T16"
120 LPRINT CHR$(27);"N"
130 FOR N = 1 TO 8
140   LPRINT "MSXBASIC"
150   LPRINT CHR$(27);"S0064";
160   FOR M = 1 TO 64
170     LPRINT CHR$(2^(M MOD 8)-1);
180   NEXT M
190   LPRINT
200 NEXT N
210 END
```

6.3.2 グラフィック印字のモード

グラフィック印字には以下のモードがあります。

1. ノーマルモード（ESC M）

   電源ON直後はこのモードになり、他のモードからはESC Mでこのモードに戻ります。このモードではドットピッチなどもなるべく他のプリンタと同じになるようにします。

2. コピーモード（ESC D）

   ESC Dでこのモードが選択されます。このモードでは画面コピーなどで最も多く使用される使用方法を想定してグラフィック印字を行います。したがって、ドットピッチなどは条件を満たす範囲で変更してかまいません。

3. コンパチブルモード（SUB A、SUB B、SUB C、SUB D）

SUBシーケンスによりこのモードが選択されます。このモードでは、いろいろな場合に対応した印字方法が選べます。このモードではドットピッチなども機種に依存せずに規定されます。

6.3.3 ノーマルモード（ESC M）

このモードではドットピッチなどもなるべく他のプリンタと同じになるようにします。

1. 8ドットグラフィックス（ESC S、ESC V）

使用目的	特に規定しません。		
達成目標	縦ピッチ	横ピッチ	横方向最大印字ドット数
	1/60インチ	1/80インチ	640（パイカ）
	1/60インチ	1/160インチ	1280（プロポーショナル）
	最大印字ドット数（1280ドット）を優先します。		

2. 16ドットグラフィックス（ESC I、ESC W）

使用目的	特に規定しません。		
達成目標	縦ピッチ	横ピッチ	横方向最大印字ドット数
	1/120インチ	1/160インチ	1280
	縦ピッチ、横ピッチともなるべく16ドットプリンタに近くします。		

3. 24ドットグラフィック印字（ESC J、ESC U）

仕様	縦ピッチ	横ピッチ	横方向最大印字ドット数
	1/180インチ	1/180インチ	1440

6.3.4 コピーモード

このモードではスクリーンコピーなどで最も使用される可能性の高い使用方法を想定し、その使用方法だけをサポートすることを前提として、プリンタの特性にあわせて、最も美しいイメージでグラフィック印字を行います。

1. 8ドットグラフィックス（ESC S、ESC V）

使用目的	画面コピーなど		
達成目標	縦ピッチ	横ピッチ	横方向最大印字ドット数
	1/60インチ	1/80インチ	640（パイカ）
	1/60インチ	1/160インチ	1280（プロポーショナル）
横ピッチ	10インチを優先する。		

2. 16ドットグラフィックス（ESC I、ESC W）

使用目的	グラフィック印字など。印字の美しさを要求するもの。		
仕様	縦ピッチ	横ピッチ	横方向最大印字ドット数
	1/180インチ	1/180インチ	1440

このモードでは必ずこの印字方法をとって下さい。

3. 24ドットグラフィック印字（ESC J、ESC U）

ノーマルモードと同じ。

6.3.5 コンパチブルモード

このモードはノーマルモード、コピーモードで対応できない場合に使用します。このモードをサポートする場合は使用するアプリケーションに合ったモードをディップスイッチなどで選択できる必要があります。
印字方法を以下に示します。

ヘッドピンNO. タイプ	縦8ドットビットイメージ (1)	縦8ドットビットイメージ (2)	縦16ドットビットイメージ (3)	縦16ドットビットイメージ (4)	縦24ドットビットイメージ (5)
# 1	d1	d1	d1	d1	d1
# 2	d1		d1 d2	d2	d2
# 3		d2	d2	d3	d3
# 4	d2		d3	d4	d4
# 5	d2	d3	d3 d4	d5	d5
# 6			d4	d6	d6
# 7	d3	d4	d5	d7	d7
# 8	d3		d5 d6	d8	d8
# 9		d5	d6	d9	d9
# 10	d4		d7	d10	d10
# 11	d4	d6	d7 d8	d11	d11
# 12			d8	d12	d12
# 13	d5	d7	d9	d13	d13
# 14	d5		d9 d10	d14	d14
# 15		d8	d10	d15	d15
# 16	d6		d11	d16	d16
# 17	d6		d11 d12		d17
# 18			d12		d18
# 19	d7		d13		d19
# 20	d7		d13 d14		d20
# 21			d14		d21
# 22	d8		d15		d22
# 23	d8		d15 d16		d23
# 24			d16		d24

OR処理

イメージデータ

	MSB							LSB
D1	8	7	6	5	4	3	2	1

	MSB							LSB
D1	8	7	6	5	4	3	2	1
D2	16	15	14	13	12	11	10	9

	MSB							LSB
D1	8	7	6	5	4	3	2	1
D2	16	15	14	13	12	11	10	9
D3	24	23	22	21	20	19	18	17

印字方法	モード
8ドットイメージ（ESC S、ESC V）	
縦24ドット使用、縦ドットピッチ1/60インチ	
720ドット/8インチ	SUB Dモード
960ドット/8インチ	SUB Aモード
1280ドット/8インチ	SUB Bモード
1440ドット/8インチ	SUB Cモード
16ドットイメージ（ESC I、ESC W）	
縦24ドット使用、縦ドットピッチ1/120インチ相当	
960ドット/8インチ	SUB Aモード
1280ドット/8インチ	SUB Bモード
1440ドット/8インチ	SUB Cモード
縦16ドット使用、縦ドットピッチ1/180インチ相当	
720ドット/8インチ	SUB Dモード

6.3.6 コピーモード、ノーマルモードの印字方法の例

8ドット、16ドットビットイメージのコピーモード、ノーマルモードでの印字方法はハードウェアによりサポートできる範囲が異なるためとくに定めませんが、以下に印字方法の一例を示します。

ノーマルモード	
8ビットイメージ改行ピッチ	1/120インチ
縦ドットピッチ	1/60インチ
横ドットピッチ	1/180インチ
横方向最大印字ドット数	1440ドット
16ドットイメージ改行ピッチ	1/120インチ
縦ドットピッチ	1/120インチ
横ドットピッチ	1/180インチ
横方向最大印字ドット数	1440ドット

コピーモード	
8ドットイメージ改行ピッチ	1/180インチ
縦ドットピッチ	1/90インチ
横ドットピッチ	1/90インチ
横方向最大印字ドット数	720ドット
16ドットイメージ改行ピッチ	1/180インチ
縦ドットピッチ	1/180インチ
横ドットピッチ	1/180インチ
横方向最大印字ドット数	1440ドット

8ピン、16ピンプリンタとの互換性をさらに高めるには以下の方法をとって下さい。

1. ノーマルモードにおける8ドットイメージの印字密度を8ピンプリンタと同じように印字モードによって切り換えます。

2. ノーマルモードにおける16ドットイメージ印字の印字密度を16ドットプリンタと同じにします。

6.3.7 その他の印字方法をとる場合の注意

1. ノーマルモードの8ドットグラフィックスでは最大印字ドット数1280を保証します。

2. コピーモードの8ドットグラフィックスでは最大印字ドット数640ドットを保証し、縦横比を8ドットプリンタに近くするか、1：1にします。

3. ノーマルモードの8ドットグラフィックス印字において、ワープロなどのソフトで半ドットピッチずらして16ドット印字するものがあるので、ドット間のスペースを充分とります。

4. コピーモードの16ドットグラフィックスについてはなるべくこの方法とします。

6.4 漢字コード

MSX24ドット漢字プリンタの漢字コードはMSX標準の漢字ROMのコードに準拠します。ただし、罫線についてはMSX標準の漢字ROMで規定していませんので、1983年版JISコードに規定されている8区の罫線記号を追加することを推奨します。

6.5 縦拡大文字の網掛け処理

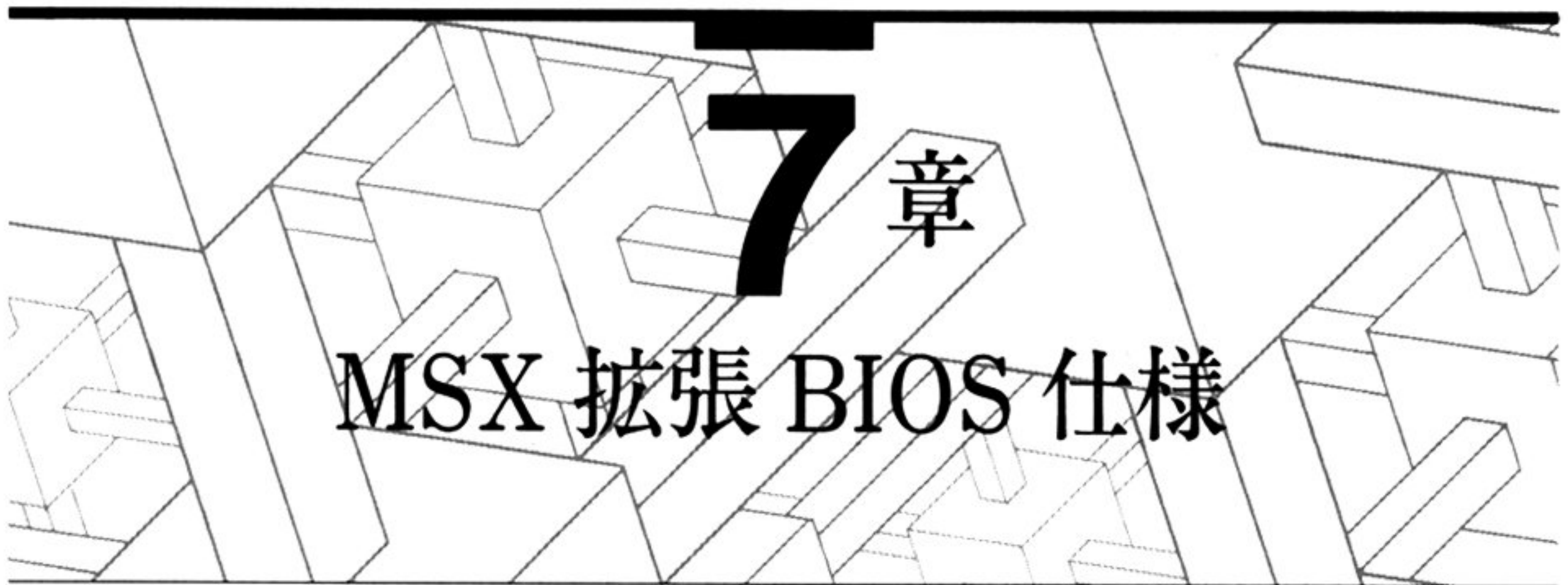
縦拡大文字に対して網掛けを行うときは、必ず以下のようにして下さい。このイメージデータは、網掛、罫線のパターンデータです。

1. イメージデータ上側
2. CR
3. 縦拡大モード設定
4. 縦拡大データ
5. 縦拡大モード解除
6. CR+LF (1/120)
7. イメージデータ下側

6.6 サンプルプログラム

MSX漢字BASICとM1024IIP/Xでの組み合わせによる、縦拡大文字に対する網掛処理の例が添付のフロッピーディスクに入っています。プログラム名は「PRINT.BAS」です。

なお、【RAWPRT (0F418H)】が0以外になっていないと制御コードなどをそのまま送出できませんので、ご注意下さい。

7章 MSX拡張BIOS仕様

7.1 概要

MSXではハードウェアを拡張する際は、そのハードウェアをカートリッジスロットに接続します。そして、そのハードウェアを制御するソフトウェアは、ROMの形でそのハードウェア上に実装されます。

もし、その制御用ソフトウェアが拡張BASICステートメントの形式であれば、ユーザーはプログラミングのときにスロットの切り換えなどを意識する必要はありません。

しかし、機械語やそれに準ずる言語(例えば、C言語など)でそのハードウェアや制御用ソフトウェアをアクセスする場合、どのスロットに呼び出し先のハードウェアが接続されているかがわからなければなりません。

この問題を解決するために定められたのが、拡張BIOSという手法です。拡張BIOSコールを使えば、

- 何の拡張ハードウェアがシステムに実装されているか
- その数はいくつか
- どのカートリッジスロットに接続されているか

などを調べることができます。

この章では、拡張BIOSコールを使用する方法と拡張BIOSそのものを作成するために必要な情報を解説します。

7.2 拡張BIOSにおけるデバイス

拡張BIOSでは、拡張ハードウェアとその制御用ソフトウェアの組み合わせを「デバイス」とし、0〜255の番号を割り当てています。違う種類のデバイスに同じ番号をつけることはできません。したがって、デバイス番号はアスキーが登録・管理しています。

デバイス番号を拡張BIOSに渡すには、呼び出しの際にDレジスタに番号を入れます。

現在登録されているデバイス番号は以下のとおりです。

表7.100　登録されているデバイス一覧

番　号	デバイス
0	全てのデバイスを意味する
1〜3	未使用
4	MSX-DOS2（メモリマッパーサポート）
5〜7	未使用
8	RS232C、MSX-MODEM
9	未使用
10	MSX-AUDIO
11	MSX-MIDI
12〜15	未使用
16	MSX-JE
17	漢字ドライバ
18〜254	未使用
255	システムエクスクルーシブ

7.3 拡張BIOSの呼び出し

ここでは、拡張BIOSの呼び出し方について説明します。

【HOKVLD(FB20H)】のビット0を調べ、「0」ならば拡張BIOSはありません。「1」ならば拡張BIOSが設定されているので、ルールに従い【EXTBIO(FFCAH)】を呼び出します。

詳しくは、各デバイスの拡張BIOSの説明をご参照下さい。

リスト7.6は、アプリケーションプログラムが拡張BIOSの「割り込み禁止の宣言」を呼び出す例です。

リスト7.6 「割り込み禁止の宣言」の使用例

```
hokvld  equ     0fb20h          ; address of Extended BIOS valid flag
extbio  equ     0ffcah          ; entry address of extended BIOS

        ld      a, (hokvld)     ; get valid flag
        rrca                    ; is the extended BIOS valid ?
        jr      nc, noextbio    ; no..
        ld      d, 0            ; broad-cast command
        ld      e, 2            ; say 'disable interrupt'
        call    extbio          ; call extended BIOS
                .
                .

;
;  Enters here when no extended BIOS
;
noextbio:
        di
        halt                    ; halt here
        jr      noextbio
```

7.4 拡張BIOSの実現方法

ここでは、拡張BIOSを実現するためのデバイス側のプログラムについて説明します。
デバイス側のソフトウェアは、まず次のような手順で環境を整えます。

1. 【HOKVLD(0FB20H)】の内容を調べ、「0」ならばこれを「1」にして、【EXTBIO(0FFCAH)】からの29バイトにC9H（RET命令）を書く。

2. 自分の後に【EXTBIO(0FFCAH)】を使用するデバイス側のプログラムのために、【EXTBIO(0FFCAH)】からの5バイトを自分のワークエリアにセーブする。
   処理の最後で、セーブしたワークエリアにジャンプすることで、複数のデバイスのデバイスが同じワークエリア(EXTBIO)を使うことができる(「INIEXBIO.MAC」参照)。

3. 自分の拡張BIOSのエントリへのインタースロットコール命令を【EXTBIO(0FFCAH)】からに書く。

4. DISINT、ENAINTのプログラムを図7.52の指定アドレスに書く。

0FFCAH
拡張 BIOS のエントリ
(5 バイト)
0FFCFH
DISINT のエントリ
(5 バイト)
0FFD4H
ENAINT のエントリ
(5 バイト)
0FFD9H
DISINT / ENAINT
プログラムエリア
(14 バイト)
0FFE6H

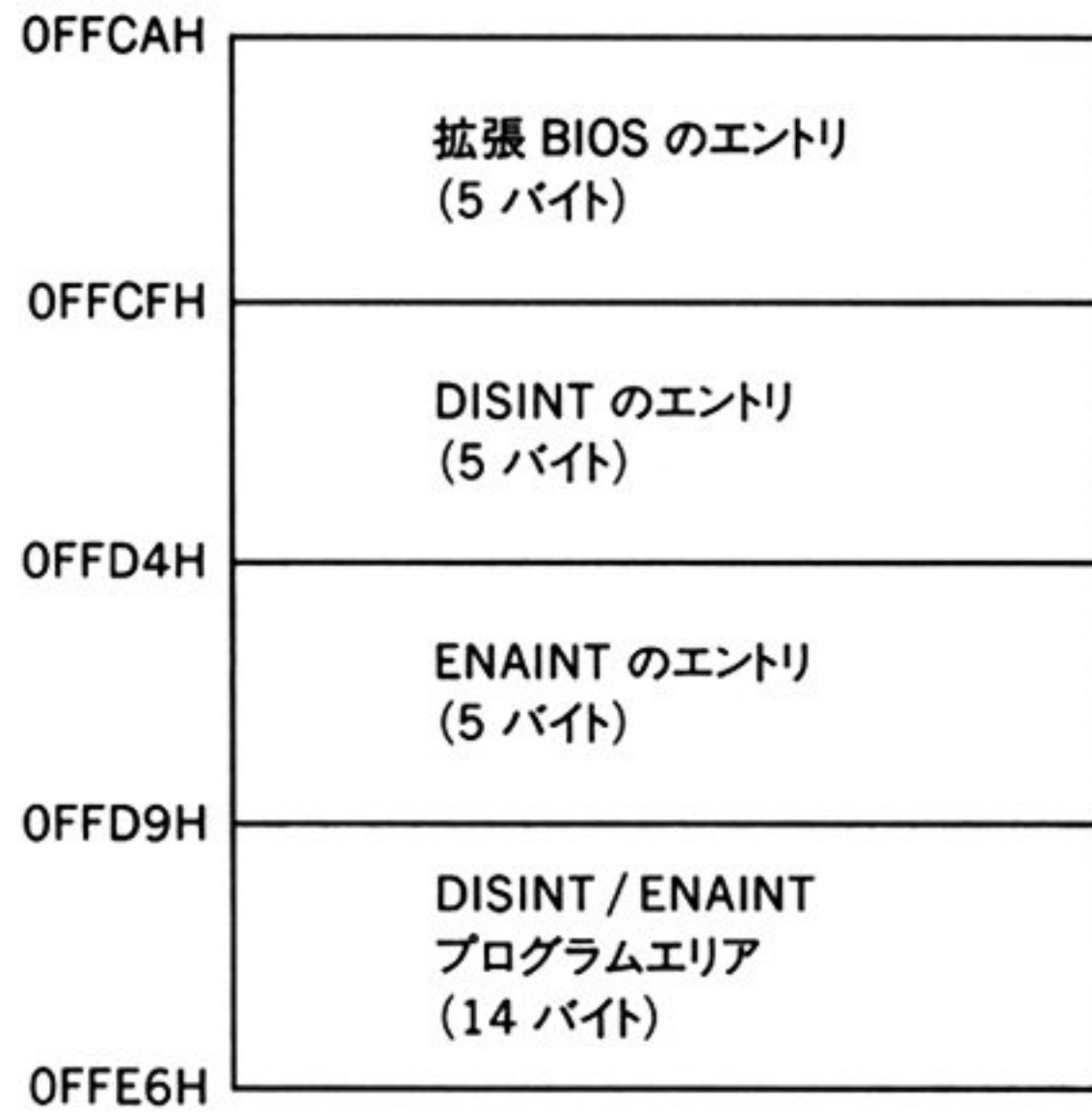



図 7.52　拡張 BIOS エントリの記憶領域

添付のフロッピーディスクに、拡張 BIOS を設定するためのサンプルプログラム「INIEXBIO.MAC」が入っていますので、参照して下さい。

このプログラム中に GETSLT というルーチンがあります。これはスロット切り換えをしないで、CPU が直接アクセスできるメモリ（メイン RAM）のスロットアドレスを得るものです。このプログラムは以降のプログラム例でも使います。

スロットアドレスとは各スロットにつけられたアドレスで、8 ビットで表されます。その形式は以下のとおりです。

7	6	5	4	3	2	1	0
F	0	0	0	S	S	P	P

基本スロット番号（0～3）
拡張スロット番号（0～3）
拡張スロットを指定するときは「1」

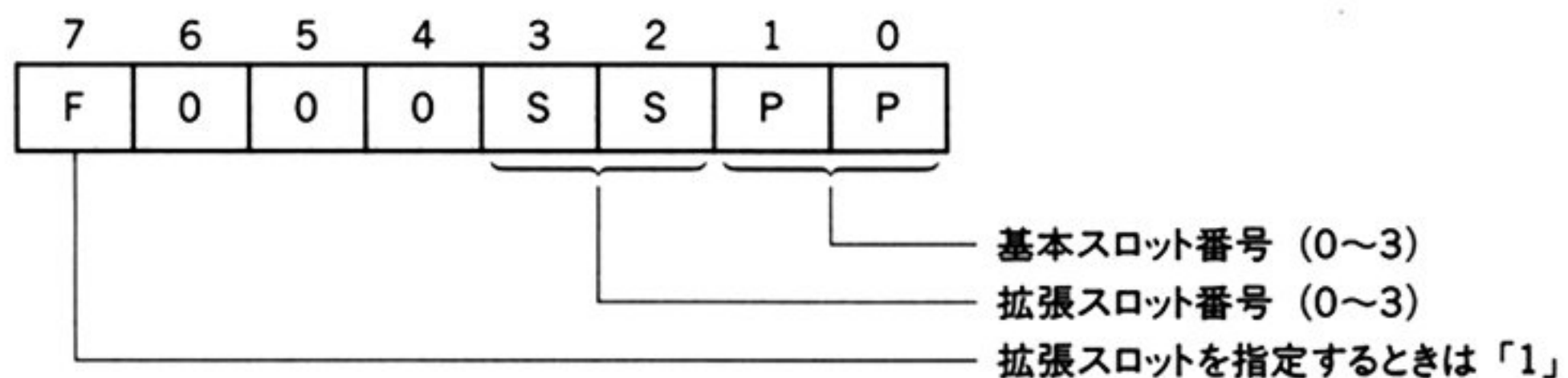



図 7.53　スロットアドレスの形式

7.5 拡張 BIOS の機能

拡張 BIOS の機能は以下の 3 つに分けられます。

1. デバイス番号 0 によるブロードキャストコマンド（すべてのデバイスの呼び出し）
2. デバイス番号による個々のデバイスの呼び出し
3. デバイス番号 255 によるシステムエクスクルーシブ(そのデバイスのメーカーが独自の拡張 BIOS を組み入れる場合)

なお、拡張 BIOS には、以下ような制限があります。

拡張 BIOS はインタースロットコールで呼び出され、しかもネストしています。そのためデバイスが 1 つ呼ばれるたびに最低 16 バイトのスタックエリアが必要になります。また、8000H〜BFFFH に拡張 BIOS のプログラムが配置されていることもあります。したがって、拡張 BIOS を呼び出す際のスタックポインタは C000H よりも上位のアドレスになければなりません。

デバイスの制御プログラム (拡張 BIOS) は、ブロードキャストコマンドと各々のデバイスの呼び出しコマンドを必ずサポートして下さい。

機能の選択は E レジスタに機能番号を入れて行います。各機能は以下で説明します。

7.5.1 ブロードキャストコマンド

デバイス番号 0 による全デバイスの選択です。ブロードキャストコマンドには、以下の機能があります。

機能番号	意　味
0	デバイス番号の取得
1	トラップ使用数の取得
2	割り込み禁止の宣言
3	割り込み許可の宣言

機能番号 0

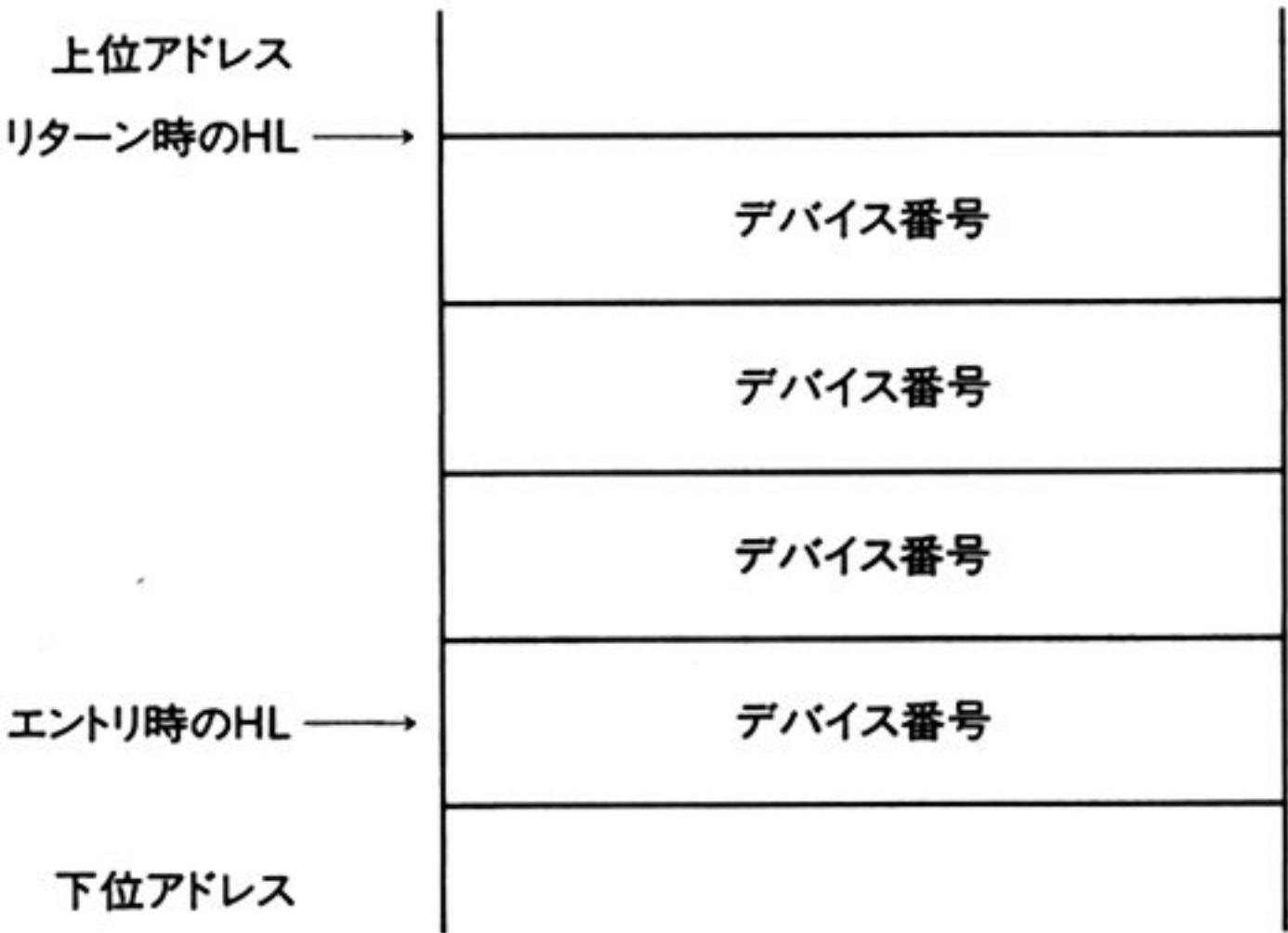
機　能

デバイス番号を取得します。

解　説

システムにどんなデバイスがあるかを調べます。各々のデバイスは呼び出し元が用意したテーブルに自分のデバイス番号を書き込んだ後、テーブルのポインタを1つインクリメントして、次のデバイスに制御を渡します。
呼び出し元に制御が戻ったとき、ポインタは最後に書き込まれたアドレス+1を指しています。
テーブルのポインタはBレジスタとHLレジスタで指定します。Bレジスタはテーブルが存在するメモリのスロットアドレスで、HLレジスタはメモリアドレスです。

■テーブルの例（デバイスが4つ存在している場合）

上位アドレス	
リターン時のHL →	デバイス番号
	デバイス番号
	デバイス番号
エントリ時のHL →	デバイス番号
下位アドレス	

リスト7.7は、アプリケーションプログラムがブロードキャストコマンドの「デバイス番号の取得」を呼び出す例です。

リスト7.7　デバイス番号の取得例

```
;
;  GETDEV   - Get device number
;  Entry    : none
;  Return   : carry flag is set if no devices
;  Modify   : all
;
```

```
extbio  equ     0ffcah          ; entry address of extended BIOS

getdev:
        ld      bc, table
        call    getslt          ; get slot address of the TABLE
        ld      h, b
        ld      l, c
        ld      b, a
        ld      d, 0            ; broad-cast command
        ld      e, 0            ; 'get device number' command
        push    hl              ; save TABLE address
        call    extbio
        pop     de              ; restore TABLE address
        or      a
        sbc     hl, de          ; how many devices ?
        ret     nz              ;  there are some devices
        scf                     ;  no devices
        ret

table:
        ds      32              ; assume maximum 32 devices
```

機能番号 1

機 能

トラップ使用数を取得します。

解 説

MSX BASIC には、「ON STOP」や「ON SPRITE」などのトラップ機能があります。このエントリはトラップ機能を外部で拡張するために使用します。MSX BASIC は内部で 26 個のイベントを管理しています。その内訳は以下のとおりです。

イベント番号	用 途
0〜9	ON KEY GOSUB
10	ON STOP GOSUB
11	ON SPRITE GOSUB
12〜16	ON STRIG GOSUB
17	ON INTERVAL GOSUB
18〜23	拡張デバイス用
24〜25	システム予約（使用禁止）

18番目から23番目までを拡張デバイスに使用することができます。トラップを使用するデバイスはイニシャライズ時にこの機能を呼び出し、他のデバイスが幾つ使うかを調べて、18+（この機能の返り値）番目のトラップから使用します。
この機能を呼び出す際には。Aレジスタに「0」をセットして下さい。また、戻り値（各デバイスが使っているトラップの数）はAレジスタに入ります。
リスト7.8は、アプリケーションプログラムがブロードキャストコマンドの「トラップ使用数の取得」を呼び出す例です。

リスト7.8　トラップ使用数の取得例

```
;
; GETTRP  - get how many traps are already used
; Entry   : none
; Return  : [A] = number of traps are already used
; Modify  : [Flag]、[DE]
;
gettrp:
        xor     a               ; clear number of traps
        ld      d, 0            ; broad-cast command
        ld      e, 1            ; 'get number of traps' command
        call    extbio
        ret
```

機能番号 2

機　能

割り込みの禁止を宣言します。

解　説

CPUの割り込み機能を利用するデバイスがある場合、別のデバイスが割り込みを一定期間以上禁止すると割り込み処理に不都合が生じることがあります。例えば、RS-232Cカートリッジは受信割り込みを使用するので、別のデバイスが長期間割り込みを禁止してしまうと、受信データを取りこぼしてしまいます。
これを避けるために、長期間割り込みを禁止する場合は、その前にこのエントリを呼び出します。割り込みを禁止する時間の長さは1mS以上の場合とします（1mS以上割り込みが禁止されると9600bpsの通信で受信ができなくなる）。
呼ばれたデバイス側では、割り込みが禁止されても都合の良いように内部で処理し（RS-232CではXOFFを送るなど）戻ります。割り込みを許可するときは、次の「割り込み許可の宣言」を必ず呼び出して下さい。

リスト7.9は、アプリケーションプログラムがブロードキャストコマンドの「割り込み禁止の宣言」を呼び出す例です。

この機能には、レジスタへの設定値や戻り値はありません。

リスト7.9 割り込み禁止の宣言例

```
;
;   DISINT     - disable interrupt while long time
;   Entry      : none
;   Return     : none
;   Modify     : none
;
disint:
            ld        d, 0              ; broad-cast command
            ld        e, 2              ; 'disable interrupt' command
            call      EXTBIO
            ret
```

機能番号 3

機　能

割り込みの許可を宣言します。

解　説

機能番号2で禁止宣言されていた割り込みを許可します。

リスト7.10は、アプリケーションプログラムがブロードキャストコマンドの「割り込み許可の宣言」を呼び出す例です。

この機能には、レジスタへの設定値や戻り値はありません。

リスト7.10 割り込み許可の宣言例

```
;
;   ENAINT     - enable interrupt
;   Entry      : none
;   Return     : none
;   Modify     : none
;
disint:
            ld        d, 0              ; broad-cast command
            ld        e, 3              ; 'enable interrupt' command
            call      extbio
            ret
```

7.5.2 各デバイスに対するコマンド

各々のデバイスを選択して発行するコマンドです。各デバイスに対しては、機能番号0のみが決められており、その他の機能はデバイスによって異なります。

詳しくは、各デバイスの拡張BIOSの説明を参照して下さい。

機能番号0

機 能

エントリアドレスを取得します。

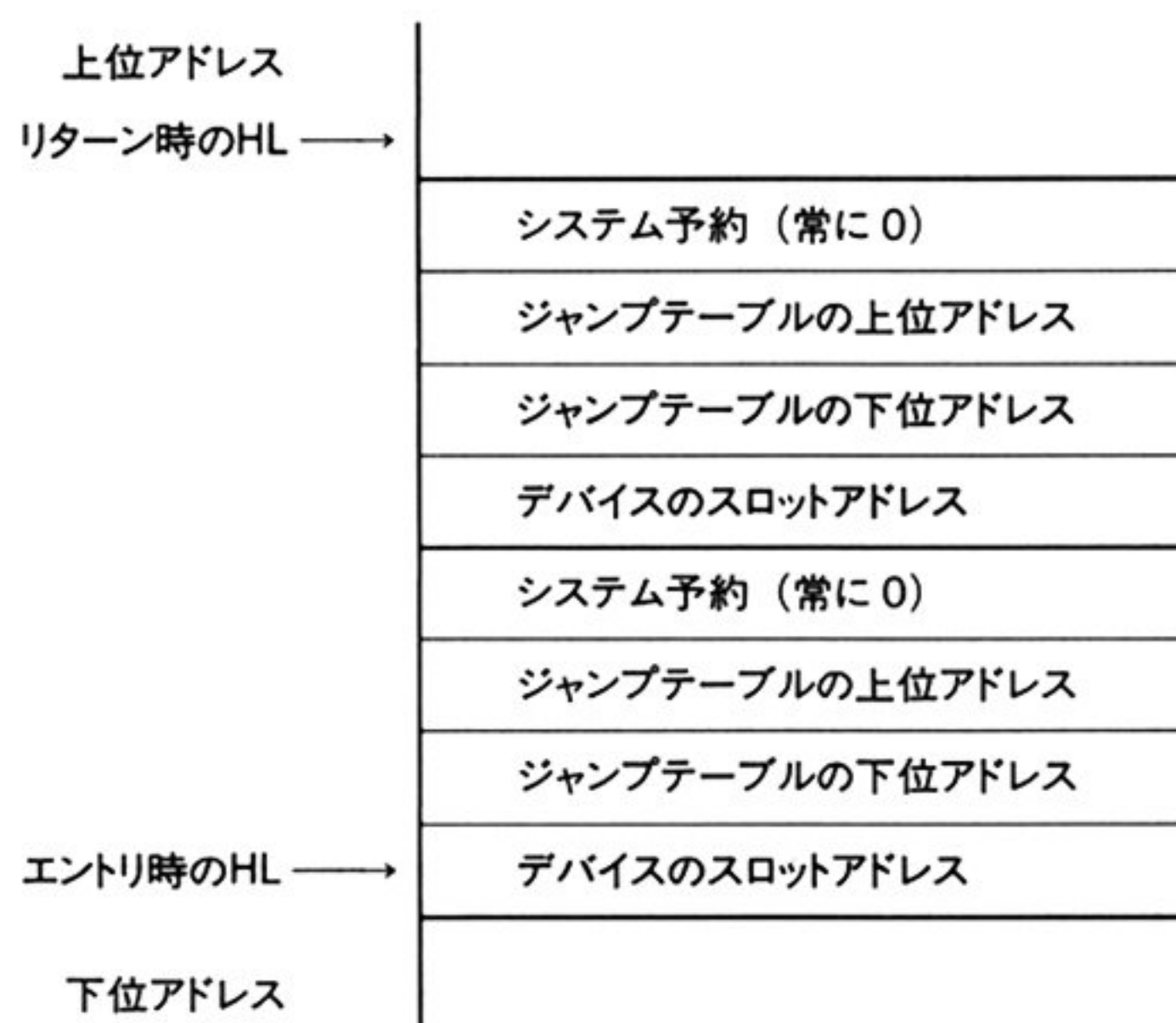
解 説

各デバイスが持っているBIOSのジャンプテーブルの先頭アドレスとスロットアドレスを調べます。各々のデバイスはデバイス番号を見て、自分が処理すべき命令かどうかを判断し、違うときは次のデバイスに制御を渡します。自分が処理すべき命令ならば、呼び出し元が用意したテーブルにジャンプテーブルの先頭アドレスとスロットアドレスを書き込んだ後、次のデバイスに制御を渡します。

呼び出し元に制御が戻ったとき、ポインタは最後に書き込まれたアドレス+1を指しています。

テーブルのポインタはBレジスタとHLレジスタで指定します。Bレジスタはテーブルのあるメモリのスロットアドレスで、HLレジスタはメモリアドレスです。

■テーブル例(同じ番号のデバイスが2つ存在している場合)

上位アドレス	
リターン時のHL ──→	
	システム予約（常に0）
	ジャンプテーブルの上位アドレス
	ジャンプテーブルの下位アドレス
	デバイスのスロットアドレス
	システム予約（常に0）
	ジャンプテーブルの上位アドレス
	ジャンプテーブルの下位アドレス
エントリ時のHL ──→	デバイスのスロットアドレス
下位アドレス	

リスト7.11は、アプリケーションプログラムが各デバイスの「エントリアドレスの取得」を呼び出す例です。

リスト7.11　各デバイスのエントリアドレスの取得

```
;
;  GETENT   - Get entry address
;  Entry    : none
;  Return   : carry flag is set if no devices
;  Modify   : all
;
device  equu    8                   ; device # 8 is RS-232C
getent:
        ld      bc, table
        call    getslt              ; get slot address of TABLE
        ld      h, b
        ld      l. c
        ld      b, a
        ld      d, device           ; set deice number
        ld      e, 0                ; 'get entry address' command
        push    hl                  ; save TABLE address
        call    extbio
        pop     de                  ; restore TABLE address
        or      a
        sbc     hl, de              ; how many devices ?
        ret     nz                  ;  there are some devices
        scf                         ;  no devices
        ret

table:
        ds      32*4                ; assume maximum 32 devices
```

7.5.3 システムエクスクルーシブ

機能番号 0

機　能

エントリアドレスを取得します。

解　説

各デバイスが持っているメーカー独自の拡張BIOSのジャンプテーブルの先頭アドレスとスロットアドレスおよびメーカーコードを調べます。各々のデバイスは、呼び出し元が用意したテーブルにジャンプテーブルの先頭アドレスとスロットアドレスおよびメーカーコードを書き込んだ後、次のデバイスに制御を渡します。呼び出し元に制御が戻ったとき、ポインタは最後に書き込まれたアドレス+1を指しています。全てのデバイスが値を返してきますので、特定のデバイスを選択するときは、「7.5.2 各デバイスに対するコマンド（機能番号0)」と併用するようにします。

テーブルのポインタはBレジスタとHLレジスタで指定します。Bレジスタはテーブルのあるメモリのスロットアドレスで、HLレジスタはメモリアドレスです。

■ テーブルの例（メーカー独自のデバイスが2つ存在している場合）

上位アドレス リターン時のHL →	
	システム予約（常に0）
	メーカーコード
	ジャンプテーブルの上位アドレス
	ジャンプテーブルの下位アドレス
	デバイスのスロットアドレス
	システム予約（常に0）
	メーカーコード
	ジャンプテーブルの上位アドレス
	ジャンプテーブルの下位アドレス
エントリ時のHL →	デバイスのスロットアドレス
下位アドレス	

リスト 7.12 は、アプリケーションプログラムがシステムエクスクルーシブの「エントリアドレスの取得」を呼び出す例です。

リスト 7.12　システムエクスクルーシブのエントリアドレスの取得例

```
;
;   SYSENT   - Get entry address
;   Entry    : none
;   Return   : carry flag is set if no devices
;   Modify   : all
;
sysent:
        ld      bc, table
        call    getslt          ; get slot address of TABLE
        ld      h, b
        ld      l, c
        ld      b, a
        ld      d, 0ffh         ; system exclucive
        ld      e, 0            ; 'get entry address' command
        push    hl              ; save TABLE address
        call    extbio
        pop     de              ; restore TABLE address
        or      a
        sbc     hl, de          ; how many devices ?
        ret     nz              ;   there are some devices
        scf                     ;   no devices
        ret

table:
        ds      32*5            ; assume maximum 32 devices
```

表 7.101 メーカーコード一覧

コード	メーカー名
0	アスキー
1	マイクロソフト
2	キヤノン
3	カシオ計算機
4	富士通
5	富士通ゼネラル
6	日立製作所
7	京セラ
8	松下電器産業
9	三菱電機
10	日本電気
11	ヤマハ
12	日本ビクター
13	フィリップス
14	パイオニア
15	三洋電機
16	シャープ
17	ソニー
18	スペクトラビデオ
19	東芝
20	ミツミ電機
21	テレマティカ
22	グラディエンテ
23	シャープドブラジル
24	GOLD STAR
25	DEAWOO
26	SumSong

APPENDIX

A.1 BIOS 一覧

ここではユーザーが利用可能な BIOS エントリを紹介します。

BIOS には MAIN ROM と SUB ROM 内のルーチンの 2 種類があり、コールの手順はそれぞれ異なります。ここでエントリは次のように表記します。

各 BIOS の名前

呼び出し番地（16 進数）

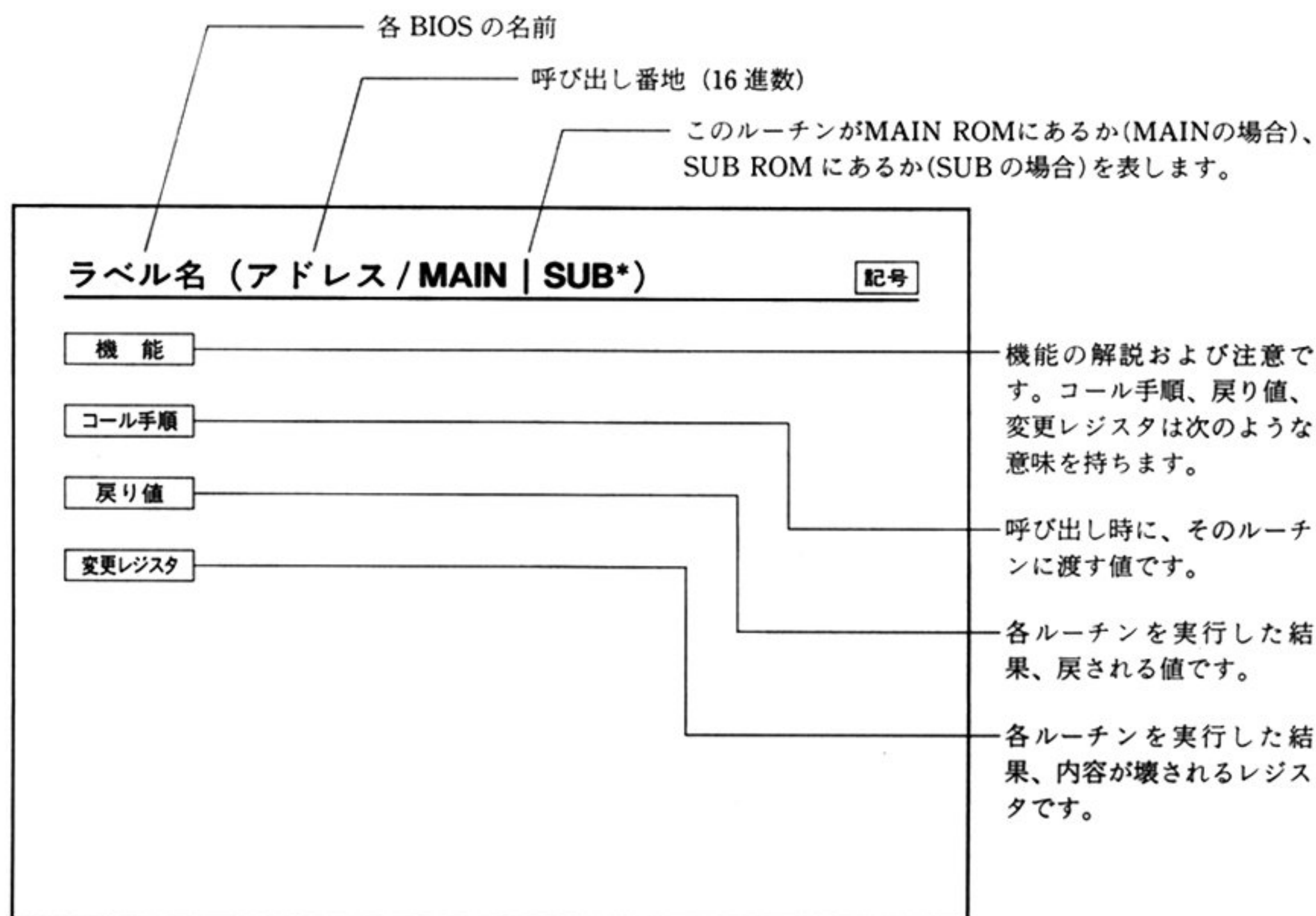
このルーチンがMAIN ROMにあるか(MAINの場合)、SUB ROM にあるか(SUB の場合)を表します。

ラベル名（アドレス / MAIN | SUB*） 記号

機能 — 機能の解説および注意です。コール手順、戻り値、変更レジスタは次のような意味を持ちます。

コール手順 — 呼び出し時に、そのルーチンに渡す値です。

戻り値 — 各ルーチンを実行した結果、戻される値です。

変更レジスタ — 各ルーチンを実行した結果、内容が壊されるレジスタです。

記号 は次のような意味を持ちます。

記　号	意　味
MSX	MSX1 からサポートされているルーチン。
MSX2	MSX2 用に追加または機能拡張されたルーチン。
MSX2＋	MSX2＋用に追加または機能拡張されたルーチン。

[A]、[HL]などレジスタ名を[　]で囲んだものは、そのレジスタの内容を示します。例えば、HL レジスタの内容が 8000H であった場合に、[HL]とは 8000H を示していることになります。

【　】で囲んだものは、ワークエリア中の名前と番地です。例えば、【PRTFLG(F416H)】とあれば、ワークエリアの F416H 番地にある PRTFLG という名前の場所を示します。「A.2 ワークエリア」をご参照ください。

A.1.1 MAIN ROM

MAIN ROM 内のルーチンを呼び出す場合は、通常のサブルーチンコールとして CALL 命令または、RST 命令で行います。

■ RST 関係

以下の RST のうち、RST 00H～RST 28H は BASIC インタプリタが使います。RST 30H はインタースロットコールに、RST 38H はハードウェア割り込みに使います。

CHKRAM（0000H / MAIN） MSX

機　能

RAM をチェックし、システム用の RAM に使うスロットを選択します。このルーチンの実行後は、さらに初期化のルーチンへ分岐します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

SYNCHR（0008H / MAIN） MSX

機　能

HL レジスタが指す文字が指定した文字かどうかを調べます。違っていたら「Syntax error」を発生し、同じであれば CHRGTR（0010H / MAIN）へジャンプします。

コール手順

HL　チェックする文字。このルーチンを呼び出すRST命令の後に比較する文字を入れる（インラインパラメータ）。

例

```
        ld      hl, moji
        rst     08h
        db      'A'
                .
                .
                .
moji:   db      'B'
```

戻り値

HL	＋1される
A	1つ進んだHLの指す文字
CY フラグ	チェックした文字が数字であればセット
Z フラグ	ステートメントの終わり（00Hまたは3AH）ならばセット

変更レジスタ

AF、HL

RDSLT（000CH / MAIN）　MSX

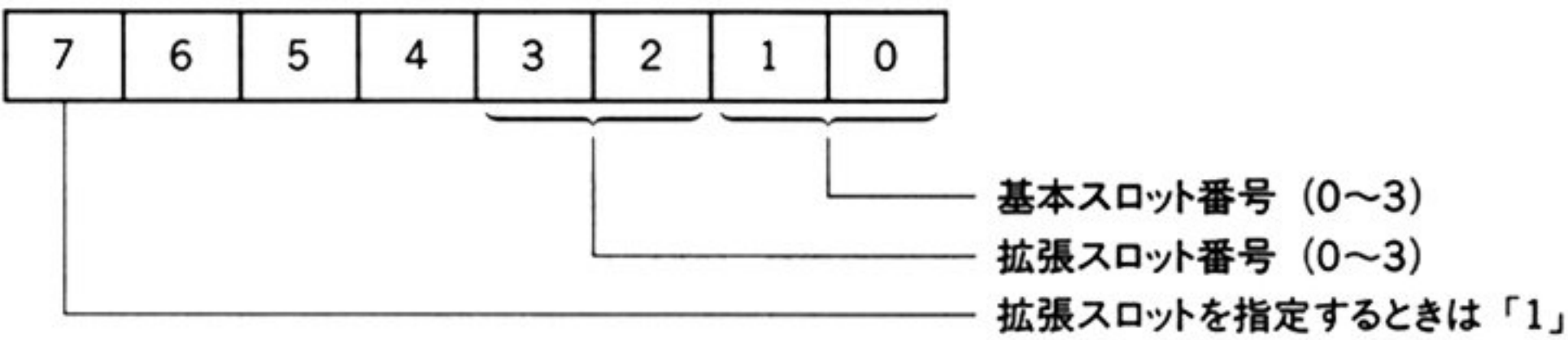
機　能

Aレジスタの値に対応するスロットを選択し、そのスロットのメモリを1バイト読み出します。このルーチンを呼ぶと、割り込みを禁止し、実行後も割り込みは解除されません。

コール手順

A　スロット番号

7	6	5	4	3	2	1	0

- ビット1～0：基本スロット番号（0～3）
- ビット3～2：拡張スロット番号（0～3）
- ビット7：拡張スロットを指定するときは「1」

HL　　読み込むメモリの番地

戻り値

A　　読み込んだメモリの値

変更レジスタ

AF、BC、DE

CHRGTR（0010H / MAIN） MSX

機　能

BASIC テキストから文字（またはトークン）を取り出します。

コール手順

HL　　読み込む文字が入っているメモリの番地

戻り値

HL	+1 される
A	1 つ進んだ HL の指す文字
CY フラグ	チェックした文字が数字であればセット
Z フラグ	ステートメントの終わり（00H または 3AH）ならばセット

変更レジスタ

AF、HL

WRSLT（0014H / MAIN） MSX

機　能

A レジスタの値に対応するスロットを選択し、そのスロットのメモリに値を 1 バイト書き込みます。このルーチンを呼ぶと、割り込みを禁止し、実行後も割り込みは解除されません。

コール手順

A　　スロット番号（形式は RDSLT と同じ）

HL	書き込むメモリの番地
E	書き込む値

戻り値

なし

変更レジスタ

AF、BC、D

OUTDO（0018H / MAIN） MSX

機　能

現在使っているデバイスに値を出力します。

コール手順

A	出力する値
【PRTFLG(F416H)】	0以外であれば、プリンタに出力
【PTRFIL(F864H)】	0以外であれば、PTRFILで示されるファイルへ出力

戻り値

なし

変更レジスタ

なし

CALSLT（001CH / MAIN） MSX

機　能

他のスロットのルーチンを呼び出します（インタースロットコール）。

コール手順

IY	上位8ビットにスロット番号（形式はRDSLTと同じ）
IX	コールする番地

戻り値

呼び出すルーチンによる

変更レジスタ

呼び出すルーチンによる

DCOMPR（0020H / MAIN） MSX

機　能

HL レジスタと DE レジスタの内容を比較します。

コール手順

HL　比較する値 1
DE　比較する値 2

戻り値

Z フラグ　HL = DE ならばセット
CY フラグ　HL<DE ならばセット

変更レジスタ

AF

ENASLT（0024H / MAIN） MSX

機　能

A レジスタの値に対応するスロットを選択し、以降そのスロットを使用可能にします。このルーチンを呼ぶと、割り込みを禁止し、実行後も割り込みは解除されません。

コール手順

A　スロット番号（形式は RDSLT と同じ）
HL　呼び出すアドレス

戻り値

なし

変更レジスタ

すべて

GETYPR（0028H / MAIN） MSX

機　能

DAC（デシマルアキュムレータ）の型を調べます。

コール手順

なし（【VALTYP(F663H)】にはDACの型が入っている）

戻り値

DACの型によって、S、Z、P/V、CYフラグが以下のように変化する

整数型	単精度実数型	文字型	倍精度実数型
C = 1	C = 1	C = 1	C = 0*
S = 1*	S = 0	S = 0	S = 0
Z = 0	Z = 0	Z = 1*	Z = 0
P/V = 1	P/V = 0*	P/V = 1	P/V = 1

各型は、*のついたフラグを調べればチェックできる

変更レジスタ

AF

CALLF（0030H / MAIN） MSX

機　能

別のスロットにあるルーチンを呼び出します。

コール手順

```
rst     30h
db      n       ; n はスロット番号（形式は RDSLT と同じ）
dw      nn      ; nn は呼び出すアドレス
```

戻り値

呼び出すルーチンによる

変更レジスタ

AF、その他は呼び出すルーチンによる

KEYINT (0038H / MAIN) MSX

機　能

タイマ割り込みの処理ルーチンを実行します。

コール手順

なし

戻り値

なし

変更レジスタ

なし

■ I/O の初期化

INITIO (003BH / MAIN) MSX

機　能

デバイスを初期化します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

INIFNK（003EH / MAIN） MSX

機　能

ファンクションキーの内容を初期化します。このルーチンを実行後、画面をクリアするとファンクションキーの表示も変わります。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

■ VDP のアクセス

DISSCR（0041H / MAIN） MSX

機　能

画面表示を禁止します。

コール手順

なし

戻り値

なし

変更レジスタ

AF、BC

ENASCR（0044H / MAIN） MSX

機　能

画面を表示します。

コール手順

なし

戻り値

なし

変更レジスタ

AF、BC

WRTVDP（0047H / MAIN） MSX2

機　能

VDPのレジスタに値を書き込みます。

コール手順

C　　VDPのレジスタ番号（レジスタ番号は0〜23、32〜46）
B　　書き込む値

戻り値

なし

変更レジスタ

AF、BC

RDVRM（004AH / MAIN） MSX

機　能

VRAMの指定したアドレスの内容を読み出します。ただし、このルーチンはTMS9918(MSX1のVDP)に対するもので、VRAMのアドレスは下位14ビットのみが有効です。全ビットを使うときは、NRDVRM（0174H / MAIN）を使います。

コール手順

HL　　VRAMのアドレス

戻り値

A　　読み出した値

変更レジスタ

AF

WRTVRM（004DH / MAIN）　MSX

機　能

VRAM にデータを書き込みます。ただし、このルーチンは TMS9918 に対するもので、VRAM のアドレスは下位 14 ビットのみが有効です。全ビットを使うときは、NWRVRM（0177H / MAIN）を使います。

コール手順

HL　　VRAM のアドレス
A　　書き込む値

戻り値

なし

変更レジスタ

AF

SETRD（0050H / MAIN）　MSX

機　能

VDP に VRAM アドレスをセットして、読み出せる状態にします。このルーチンは VDP のアドレスオートインクリメントの機能を使って、連続した VRAM 領域からデータを読み出すときに使います。このルーチンの実行後は、ポートから直接 VRAM を読み出します。したがって、RDVRM をループ中で使うより高速な読み出しができます。ただし、このルーチンは TMS9918 に対するもので、VRAM のアドレスは下位 14 ビットのみが有効です。全ビットを使うときは、NSETRD（016EH / MAIN）を使います。

コール手順

HL　　VRAMアドレス

戻り値

なし

変更レジスタ

AF

SETWRT（0053H / MAIN）

MSX

機　能

VDPにVRAMアドレスをセットして、書き込める状態にします。使用目的はSETRDと同じです。ただし、このルーチンはTMS9918に対するもので、VRAMのアドレスは下位14ビットのみが有効です。全ビットを使うときは、NSTWRT（0171H / MAIN）を使います。

コール手順

HL　　VRAMアドレス

戻り値

なし

変更レジスタ

AF

FILVRM（0056H / MAIN）

MSX2

機　能

VRAMの指定領域を同一のデータで埋めます。ただし、このルーチンはTMS9918に対するもので、VRAMのアドレスは下位14ビットのみが有効です。全ビットを使うときは、BIGFIL（016BH / MAIN）を使います。

コール手順

HL　　書き込みを開始する VRAM アドレス
BC　　書き込む領域の長さ（バイト数）
A　　書き込む値

戻り値

なし

変更レジスタ

AF、BC

LDIRMV（0059H / MAIN）　MSX2

機　能

VRAM からメモリへデータをブロック転送します。

コール手順

HL　　転送元の VRAM アドレス（指定する VRAM のアドレスは全ビットが有効）
DE　　転送先の RAM アドレス
BC　　転送する長さ（バイト数）

戻り値

なし

変更レジスタ

すべて

LDIRVM（005CH / MAIN）　MSX2

機　能

メモリから VRAM へデータをブロック転送します。

コール手順

HL　　転送元の RAM アドレス

DE　　転送先のVRAMアドレス（指定するVRAMのアドレスは全ビットが有効）
BC　　転送する長さ（単位はバイト）

戻り値

なし

変更レジスタ

すべて

CHGMOD（005FH / MAIN）　MSX2

機　能

スクリーンモードを変えます。パレットは初期化しません。パレットの初期化が必要なときは、CHGMDP（00D1H / SUB）を使います。

コール手順

A　　スクリーンモード（0〜8）

戻り値

なし

変更レジスタ

すべて

CHGCLR（0062H / MAIN）　MSX

機　能

画面の色を変えます。

コール手順

【FORCLR(F3E9H)】　前景色
【BAKCLR(F3EAH)】　背景色
【BDRCLR(F3EBH)】　周辺色

戻り値

なし

変更レジスタ

すべて

NMI（0066H / MAIN） MSX

機　能

NMI（Non Maskable Interrupt）処理ルーチンを実行します。

コール手順

なし

戻り値

なし

変更レジスタ

なし

CLRSPR（0069H / MAIN） MSX2

機　能

すべてのスプライトを次のように初期化します。

スプライトパターン	ヌル
スプライト番号	スプライト面番号
スプライトカラー	前景色
スプライトの垂直位置(SCREEN 0～3)	209
スプライトの垂直位置(SCREEN 4～12)	217

コール手順

なし

戻り値

なし

変更レジスタ

すべて

INITXT（006CH / MAIN） MSX2

機　能

画面を TEXT1 モード（SCREEN 0、40×24）に初期化します。このルーチンはパレットを初期化しません。パレットの初期化が必要であれば、このルーチンを実行した後、INIPLT（0141H / SUB）を実行します。

コール手順

【TXTNAM(F3B3H)】　パターンネームテーブルのアドレス
【TXTCGP(F3B7H)】　パターンジェネレータテーブルのアドレス
【LINL40(F3AEH)】　1行の幅（WIDTH 文によって設定する値）

戻り値

なし

変更レジスタ

すべて

INIT32（006FH / MAIN） MSX2

機　能

画面を TEXT2 モード（SCREEN 1、32×24）に初期化します。このルーチンはパレットを初期化しません。パレットの初期化が必要であれば、このルーチンを実行した後、INIPLT（0141H / SUB）を実行します。

コール手順

【T32NAM(F3BDH)】　パターンネームテーブルのアドレス
【T32COL(F3BFH)】　カラーテーブルのアドレス

【T32CGP(F3C1H)】　パターンジェネレータテーブルのアドレス
【T32ATR(F3C3H)】　スプライトアトリビュートテーブルのアドレス
【T32PAT(F3C5H)】　スプライトジェネレータテーブルのアドレス
【LINL32(F3AFH)】　1行の幅（WIDTH文によって設定する値）

戻り値

なし

変更レジスタ

すべて

INIGRP（0072H / MAIN）　MSX2

機　能

画面をGRAPHIC1モード（SCREEN 2）に初期化します。このルーチンはパレットを初期化しません。パレットの初期化が必要であれば、このルーチンを実行した後、INIPLT（0141H / SUB）を実行します。

コール手順

【GRPNAM(F3C7H)】　パターンネームテーブルのアドレス
【GRPCOL(F3C9H)】　カラーテーブルのアドレス
【GRPCGP(F3CBH)】　パターンジェネレータテーブルのアドレス
【GRPATR(F3CDH)】　スプライトアトリビュートテーブルのアドレス
【GRPPAT(F3CFH)】　スプライトジェネレータテーブルのアドレス

戻り値

なし

変更レジスタ

すべて

INIMLT（0075H / MAIN） MSX2

機　能

画面を MULTI COLOR モード（SCREEN 3）に初期化します。このルーチンはパレットを初期化しません。パレットの初期化が必要であれば、このルーチンを実行した後、INIPLT（0141H / SUB）を実行します。

コール手順

【MLTNAM(F3D1H)】　パターンネームテーブルのアドレス
【MLTCOL(F3D3H)】　カラーテーブルのアドレス
【MLTCGP(F3D5H)】　パターンジェネレータテーブルのアドレス
【MLTATR(F3D7H)】　スプライトアトリビュートテーブルのアドレス
【MLTPAT(F3D9H)】　スプライトジェネレータテーブルのアドレス

戻り値

なし

変更レジスタ

すべて

SETTXT（0078H / MAIN） MSX2

機　能

VDP のみを TEXT1 モード（SCREEN 0、40×24）にします。

コール手順

INITXT と同じ

戻り値

なし

変更レジスタ

すべて

SETT32（007BH / MAIN） MSX2

機　能

VDP のみを TEXT2 モード（SCREEN 1、32×24）にします。

コール手順

INIT32 と同じ

戻り値

なし

変更レジスタ

すべて

SETGRP（007EH / MAIN） MSX2

機　能

VDP のみを GRAPHIC1 モード（SCREEN 2）にします。

コール手順

INIGRP と同じ

戻り値

なし

変更レジスタ

すべて

SETMLT（0081H / MAIN） MSX2

機　能

VDP のみを MULTI COLOR モード（SCREEN 3）にします。

コール手順

INIMLT と同じ

戻り値

なし

変更レジスタ

すべて

CALPAT（0084H / MAIN） MSX

機　能

スプライトジェネレータテーブルの開始アドレスを獲得します。

コール手順

A　　スプライト番号

戻り値

HL　　アドレス

変更レジスタ

AF、DE、HL

CALATR（0087H / MAIN） MSX

機　能

スプライトアトリビュートテーブルの開始アドレスを獲得します。

コール手順

A　　スプライト番号

戻り値

HL　　アドレス

変更レジスタ

AF、DE、HL

GSPSIZ（008AH / MAIN） MSX

機　能

現在のスプライトサイズを獲得します。

コール手順

なし

戻り値

A	スプライトサイズ（バイト数）
CY フラグ	16×16 のサイズの場合のみセットし、それ以外のときはリセット

変更レジスタ

AF

GRPPRT（008DH / MAIN） MSX2

機　能

グラフィック画面に文字を表示します。

コール手順

A	キャラクタコード
【LOGOPR(FB02H)】	スクリーンモードが 5〜12 ならロジカルオペレーションコード

戻り値

なし

変更レジスタ

なし

■ PSGのアクセス

GICINI（0090H / MAIN） MSX

機　能

PSGを初期化し、PLAY文のための初期値を設定します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

WRTPSG（0093H / MAIN） MSX

機　能

PSGのレジスタにデータを書き込みます。

コール手順

A　　PSGのレジスタ番号
E　　書き込むデータ

戻り値

なし

変更レジスタ

なし

RDPSG（0096H / MAIN）

MSX

機　能

PSG レジスタの値を読み出します。

コール手順

A　　　PSG のレジスタ番号

戻り値

A　　　読み出した値

変更レジスタ

なし

STRTMS（0099H / MAIN）

MSX

機　能

バックグラウンドタスクとして、PLAY 文が実行中であるかどうかをチェックして、実行中でなければ音楽の演奏を始めます。

コール手順

【QUEUE(F3F3H)】の示すアドレスに、中間言語に変換された MML データをセット(「第 2 部 6.1 PLAY 文 BIOS」参照)

戻り値

なし

変更レジスタ

すべて

■キーボード、CRT、プリンタへの入出力

CHSNS（009CH / MAIN）　MSX

機　能

キーボードバッファの状態を調べます。

コール手順

なし

戻り値

Zフラグ　　バッファが空であればセット、そうでなければリセット

変更レジスタ

AF

CHGET（009FH / MAIN）　MSX

機　能

文字を1文字入力（入力待ちあり）します。

コール手順

なし

戻り値

A　　入力された文字コード

変更レジスタ

AF

CHPUT (00A2H / MAIN)

MSX

機　能

文字を1文字表示します。

コール手順

A　　出力する文字コード

戻り値

なし

変更レジスタ

なし

LPTOUT (00A5H / MAIN)

MSX

機　能

プリンタに1文字出力します。

コール手順

A　　出力する文字コード

戻り値

CY フラグ　　失敗したときセット

変更レジスタ

F

LPTSTT (00A8H / MAIN)

MSX

機　能

プリンタの状態をチェックします。

コール手順

なし

戻り値

A	255	ZフラグがリセットされていればプリンタはREADY
	0	ZフラグがセットされていればプリンタはNOT READY

変更レジスタ

AF

CNVCHR (00ABH / MAIN) MSX

機　能

グラフィックヘッダバイトかどうかをチェックします。

コール手順

A　　チェックする文字コード

戻り値

CYフラグがリセット	グラフィックヘッダではない
CYフラグとZフラグがセット	グラフィックキャラクタコードである（Aレジスタには変換後のコードが入る）
CYフラグがセット、Zフラグがリセット	グラフィックキャラクタではない（Aレジスタには渡したコードがそのまま残る）

変更レジスタ

AF

PINLIN (00AEH / MAIN) MSX

機　能

リターンキーや STOP キーがタイプされるまで、入力された文字コードをバッファに格納します。

コール手順

なし

戻り値

HL	バッファの先頭アドレス-1
CY フラグ	STOP キーで終了したときのみセット

変更レジスタ

すべて

INLIN（00B1H / MAIN） MSX

機 能

【AUTFLG(F6AAH)】がセットされる以外は PINLIN と同じ。

コール手順

なし

戻り値

HL	バッファの先頭アドレス-1
CY フラグ	STOP キーで終了したときのみセット

変更レジスタ

すべて

QINLIN（00B4H / MAIN） MSX

機 能

「?」とスペース 1 個を表示して、INLIN を実行します。

コール手順

なし

戻り値

HL　　バッファの先頭アドレス-1

CY フラグ　　STOP キーで終了したときのみセット

変更レジスタ

すべて

BREAKX（00B7H / MAIN）　MSX

機　能

CTRL ＋ STOP キーを押しているかどうかをチェックします。このルーチンは割り込みが禁止された状態でコールして下さい。

コール手順

なし

戻り値

CY フラグ　　押されていればセット

変更レジスタ

AF

BEEP（00C0H / MAIN）　MSX2

機　能

ブザーを鳴らします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

CLS（00C3H / MAIN） MSX2

機　能

画面をクリアします。

コール手順

なし

戻り値

なし

変更レジスタ

AF、BC、DE

POSIT（00C6H / MAIN） MSX

機　能

カーソルを移動します。

コール手順

H　　カーソルのX座標
L　　カーソルのY座標

戻り値

なし

変更レジスタ

AF

FNKSB（00C9H / MAIN） MSX

機　能

ファンクションキーの表示がアクティブかどうかをチェックし、アクティブなら表示し、そうでなければ消します。

コール手順

【FNKFLG(FBCEH)】

戻り値

なし

変更レジスタ

すべて

ERAFNK (00CCH / MAIN)　MSX

機　能

ファンクションキーの表示を消します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

DSPFNK (00CFH / MAIN)　MSX2

機　能

ファンクションキーを表示します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

TOTEXT（00D2H / MAIN） MSX

機　能

画面を強制的にテキストモードにします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

■汎用入出力ポートのアクセス

GTSTCK（00D5H / MAIN） MSX

機　能

ジョイスティックまたはカーソルキーの状態を調べます。

コール手順

A　　調べるジョイスティックの番号（0＝カーソルキー、1～2＝ジョイスティック）

戻り値

A　　ジョイスティックまたはカーソルキーの押された方向

0＝どの方向にも向いていない
1＝上、2＝右上、3＝右、4＝右下
5＝下、6＝左下、7＝左、8＝左上

変更レジスタ

すべて

GTTRIG（00D8H / MAIN） MSX

機　能

トリガボタンの状態を調べます。

コール手順

A　　調べるトリガボタンの番号（0＝スペースキー、1～2＝トリガボタン）

戻り値

A　　0　　トリガボタンは押されていない
　　　FFH　トリガボタンは押されている

変更レジスタ

AF

GTPAD（00DBH / MAIN） MSX

機　能

各種入出力装置の状態を調べます。

コール手順

A　　調べる入出力装置の番号

0～3	タッチパネル 1
4～7	タッチパネル 2
8～11	ライトペン
12～15	マウス 1 またはトラックボール 1
16～19	マウス 2 またはトラックボール 2

戻り値

A　　値（「第 6 部 5.4 タッチパネル、ライトペン、マウス、トラックボールの使用法」参照）

変更レジスタ

すべて

GTPDL（00DEH / MAIN） MSX2

機　能

パドルの状態を調べます。

コール手順

A　　パドルの番号（1〜12）

戻り値

A　　パドルの回転角（0〜255）

変更レジスタ

すべて

■カセット入出力ポートのアクセス

TAPION（00E1H / MAIN） MSX

機　能

カセットレコーダーのモーターを動かし、テープのヘッダブロックを読み出します。

コール手順

なし

戻り値

CY フラグ　　失敗したらセット

変更レジスタ

すべて

TAPIN（00E4H / MAIN） MSX

機　能

テープからデータを読み出します。

コール手順

なし

戻り値

A	読み込んだ値
CY フラグ	失敗したらセット

変更レジスタ

なし

TAPIOF (00E7H / MAIN) MSX

機　能

テープからの読み込みを停止します。

コール手順

なし

戻り値

A	読み込んだ値

変更レジスタ

なし

TAPOON (00EAH / MAIN) MSX

機　能

カセットレコーダーのモーターを動かし、テープのヘッダブロックを書き込みます。

コール手順

A	0	ショートヘッダ
	≠0	ロングヘッダ

戻り値

CY フラグ　　　失敗したらセット

変更レジスタ

すべて

TAPOUT（00EDH / MAIN） MSX

機　能

テープにデータを書き込みます。

コール手順

A　　　データ

戻り値

CY フラグ　　　失敗したらセット

変更レジスタ

すべて

TAPOOF（00F0H / MAIN） MSX

機　能

テープへの書き込みを停止します。

コール手順

なし

戻り値

なし

変更レジスタ

なし

STMOTR (00F3H / MAIN) MSX

機　能

カセットレコーダーのモーターの動作を設定します。

コール手順

A	0	ストップ
	1	スタート
	FFH	現在と逆の動作状態にする

戻り値

なし

変更レジスタ

AF

■その他

CHGCAP (0132H / MAIN) MSX

機　能

CAPS ランプの状態を変えます。

コール手順

A	0	ランプをつける
	≠0	ランプを消す

戻り値

なし

変更レジスタ

AF

CHGSND (0135H / MAIN) MSX

機　能

1ビットサウンドポートの状態を変えます。

コール手順

A　0　サウンドポートのビットをOFF
　≠0　サウンドポートのビットをON

戻り値

なし

変更レジスタ

AF

RSLREG (0138H / MAIN) MSX

機　能

基本スロット選択レジスタに出力している内容を読み出します。

コール手順

なし

戻り値

A　読み込んだ値

変更レジスタ

A

WSLREG (013BH / MAIN) MSX

機　能

基本スロット選択レジスタにデータを書き出します。

コール手順

A　　書き込む値

戻り値

なし

変更レジスタ

なし

RDVDP（013EH / MAIN）　MSX

機　能

VDP のステータスレジスタを読み出します。このルーチンは TMS9918 に対するものです。

コール手順

なし

戻り値

A　　読み込んだ値

変更レジスタ

A

SNSMAT（0141H / MAIN）　MSX

機　能

キーボードマトリックスから指定した行の値を読み出します。

コール手順

A　　指定する行

戻り値

A　　データ（押されているキーに対応するビットが 0 になる）

変更レジスタ

AF、C

ISFLIO（014AH / MAIN） MSX

機　能

デバイスが動作中かどうかをチェックします。

コール手順

なし

戻り値

A	0	動作中
	≠0	動作中ではない

変更レジスタ

AF

OUTDLP（014DH / MAIN） MSX

機　能

文字を1文字プリンタに出力します。LPTOUT（00A5H）とは以下の点で異なります。

1. TABはスペースに展開される。
2. MSX仕様でないプリンタに対しては、ひらがなをカタカナに、グラフィック文字を1バイト文字に変換する。
3. 失敗したときは、「device I/O error」になる。

コール手順

A　　データ

戻り値

なし

変更レジスタ

F

KILBUF（0156H / MAIN） MSX

機　能

キーボードバッファをクリアします。

コール手順

なし

戻り値

なし

変更レジスタ

HL

CALBAS（0159H / MAIN） MSX

機　能

BASICインタープリタ内のルーチンをインタースロットコールします。

コール手順

IX　　呼び出すアドレス

戻り値

呼び出すルーチンによる

変更レジスタ

呼び出すルーチンによる

■ MSX2 用に追加されたエントリ

SUBROM（015CH / MAIN） MSX2

機　能

SUB ROM をインタースロットコールします。

コール手順

IX　　呼び出すアドレス。同時に IX レジスタをスタックに積む

戻り値

呼び出すルーチンによる

変更レジスタ

裏レジスタと IY 以外

EXTROM（015FH / MAIN）

機　能

SUB ROM をインタースロットコールします。IX レジスタはスタックに積みません。

コール手順

IX　　呼び出すアドレス

戻り値

呼び出すルーチンによる

変更レジスタ

裏レジスタと IY 以外

EOL（0168H / MAIN） MSX2

機　能

行の終わりまで削除します。

コール手順

H　　X 座標
L　　Y 座標

戻り値

なし

変更レジスタ

すべて

BIGFIL（016BH / MAIN）　MSX2

機　能

機能的には FILVRM（0056H / MAIN）と同じです。以下の点が FILVRM と異なります。FILVRM では、スクリーンモードが 0～3 であるかをチェックし、もしそうなら VDP は 16K バイトの VRAM しか持っていないものとして扱います。しかし、BIGFIL はスクリーンモードのチェックを行わず、与えられたパラメータどおりに動作します。

コール手順

HL　　書き込みを開始する VRAM アドレス
BC　　書き込む領域の長さ（バイト数）
A　　書き込む値

戻り値

なし

変更レジスタ

AF、BC

NSETRD（016EH / MAIN）　MSX2

機　能

VDP にアドレスをセットして、VRAM の内容が読める状態にします。

コール手順

HL　　VRAMのアドレス

戻り値

なし

変更レジスタ

AF

NSTWRT（0171H / MAIN）　MSX2

機　能

VDPにアドレスを設定して、VRAMに書き込める状態にします。

コール手順

HL　　VRAMのアドレス

戻り値

なし

変更レジスタ

AF

NRDVRM（0174H / MAIN）　MSX2

機　能

VRAMの内容を読み出します。

コール手順

HL　　読み出すVRAMのアドレス

戻り値

A　　読み出した値

変更レジスタ

F

NWRVRM（0177H / MAIN）　MSX2

機　能

VRAMにデータを書き込みます。

コール手順

HL　書き込むVRAMのアドレス

A　書き込む値

戻り値

なし

変更レジスタ

AF

■ MSX2＋用に追加されたエントリ

RDRES（017AH / MAIN）　MSX2＋

機　能

RESETポートの内容を読み出します。

コール手順

なし

戻り値

A　MSBが0ならばハードウェアリセット

変更レジスタ

なし

WRRES（017DH / MAIN） MSX2＋

機　能

RESETポートに値を書き込みます。ハードウェアリセットをシミュレートするときは、Aレジスタの MSB を0にして、このBIOSをコールした後、BIOSの0番地にジャンプします。

コール手順

A　　書き込む値

戻り値

なし

変更レジスタ

A

A.1.2 SUB ROM

SUB ROM内のルーチンを呼び出す場合は、以下のようにインタースロットコールを呼び出して使います。

```
        .
        .
ld      ix, BIOS entry address
call    extrom
        .
        .
```

IXレジスタを壊したくない場合は、次のように呼び出します。

```
        .
        .
push    ix                  ;  save IX
ld      ix, BIOS entry address
jp      SUBROM
        .
        .
```

GRPPRT（0089H / SUB）

機　能

グラフィック画面に文字を1文字出力します（スクリーン5〜8のみで動作可能）。

コール手順

A　　文字コード

戻り値

なし

変更レジスタ

なし

NVBXLN（00C9H / SUB） MSX2

機 能

ボックスを描きます。

コール手順

BC	始点の X 座標
DE	始点の Y 座標
【GXPOS(FCB3H)】	終点の X 座標
【GYPOS(FCB5H)】	終点の Y 座標
【ATRBYT(F3F2H)】	アトリビュート（色）
【LOGOPR(FB02H)】	ロジカルオペレーションコード

戻り値

なし

変更レジスタ

すべて

NVBXFL（00CDH / SUB） MSX2

機 能

塗りつぶされたボックスを描きます。

コール手順

BC	始点の X 座標
DE	始点の Y 座標
【GXPOS(FCB3H)】	終点の X 座標
【GYPOS(FCB5H)】	終点の Y 座標
【ATRBYT(F3F2H)】	アトリビュート（色）
【LOGOPR(FB02H)】	ロジカルオペレーションコード

戻り値

なし

変更レジスタ

すべて

CHGMOD (00D1H / SUB) MSX2

機　能

スクリーンモードを変更します。

コール手順

A　　スクリーンモード (0〜8)

戻り値

なし

変更レジスタ

すべて

INITXT (00D5H / SUB) MSX2

機　能

画面をテキストモード (40×24) にして初期化します。

コール手順

【TXTNAM(F3B3H)】　パターンネームテーブルの先頭アドレス
【TXTCGP(F3B7H)】　パターンジェネレータテーブルの先頭アドレス

戻り値

なし

変更レジスタ

すべて

INIT32 (00D9H / SUB)　MSX2

機　能

画面をテキストモード（32×24）にして初期化します。

コール手順

【T32NAM(F3BDH)】　パターンネームテーブルの先頭アドレス
【T32COL(F3BFH)】　カラーテーブルの先頭アドレス
【T32CGP(F3C1H)】　パターンジェネレータテーブルの先頭アドレス
【T32ATR(F3C3H)】　スプライトアトリビュートテーブルの先頭アドレス
【T32PAT(F3C5H)】　スプライトジェネレータテーブルの先頭アドレス

戻り値

なし

変更レジスタ

すべて

INIGRP (00DDH / SUB)　MSX2

機　能

画面を高解像度グラフィックモードにして初期化します。

コール手順

【GRPNAM(F3C7H)】　パターンネームテーブルの先頭アドレス
【GRPCOL(F3C9H)】　カラーテーブルの先頭アドレス
【GRPCGP(F3CBH)】　パターンジェネレータテーブルの先頭アドレス
【GRPATR(F3CDH)】　スプライトアトリビュートテーブルの先頭アドレス
【GRPPAT(F3CFH)】　スプライトジェネレータテーブルの先頭アドレス

戻り値

なし

変更レジスタ

すべて

INIMLT（00E1H / SUB）

MSX2

機　能

画面をマルチカラーモードにして初期化します。

コール手順

【MLTNAM(F3D1H)】　パターンネームテーブルの先頭アドレス
【MLTCOL(F3D3H)】　カラーテーブルの先頭アドレス
【MLTCGP(F3D5H)】　パターンジェネレータテーブルの先頭アドレス
【MLTATR(F3D7H)】　スプライトアトリビュートテーブルの先頭アドレス
【MLTPAT(F3D9H)】　スプライトジェネレータテーブルの先頭アドレス

戻り値

なし

変更レジスタ

すべて

SETTXT（00E5H / SUB）

MSX2

機　能

VDP をテキストモード（40×24）にします。

コール手順

INITXT と同じ

戻り値

なし

変更レジスタ

すべて

SETT32（00E9H / SUB） MSX2

機　能

VDP をテキストモード（32×24）にします。

コール手順

INIT32 と同じ

戻り値

なし

変更レジスタ

すべて

SETGRP（00EDH / SUB） MSX2

機　能

VDP を高解像モードにします。

コール手順

INIGRP と同じ

戻り値

なし

変更レジスタ

すべて

SETMLT（00F1H / SUB） MSX2

機　能

VDP をマルチカラーモードにします。

コール手順

INIMLT と同じ

戻り値

なし

変更レジスタ

すべて

CLRSPR（00F5H / SUB） MSX2

機　能

すべてのスプライトを初期化します。スプライトパターンをヌルに、スプライト番号をスプライト面番号に、スプライトの色を前景色にします。スプライトの垂直位置は 217 にセットします。

コール手順

【SCRMOD(FCAFH)】　スクリーンモード

戻り値

なし

変更レジスタ

すべて

CALPAT（00F9H / SUB） MSX2

機　能

スプライトジェネレータテーブルの先頭アドレスを返します。このルーチンは MAIN ROM の同名の BIOS と同じです。

コール手順

A　　スプライト番号

戻り値

HL　　アドレス

変更レジスタ

AF、DE、HL

CALATR（00FDH / SUB）　MSX2

機　能

スプライトアトリビュートテーブルの先頭アドレスを返します。このルーチンはMAIN ROMの同名のBIOSと同じです。

コール手順

A　　スプライト番号

戻り値

HL　　アドレス

変更レジスタ

AF、DE、HL

GSPSIZ（0101H / SUB）　MSX2

機　能

現在のスプライトサイズを返します。このルーチンはMAIN ROMの同名のBIOSと同じ。

コール手順

なし

戻り値

A　　スプライトサイズ

CYフラグ　　16×16のサイズの場合のみセット

変更レジスタ

なし

GETPAT（0105H / SUB） MSX2

機　能

キャラクタパターンを返します。

コール手順

A　　文字コード

戻り値

【PATWRK(FC40H)】　キャラクタパターン

変更レジスタ

すべて

WRTVRM（0109H / SUB） MSX2

機　能

VRAM にデータを書き込みます。

コール手順

HL　　書き込む VRAM のアドレス（0〜FFFFH）

A　　書き込むデータ

戻り値

なし

変更レジスタ

AF

RDVRM （010DH / SUB） MSX2

機　能

VRAM の内容を読み出します。

コール手順

HL　　読み出す VRAM のアドレス（0～FFFFH）

戻り値

A　　読み出した値

変更レジスタ

F

CHGCLR （0111H / SUB） MSX2

機　能

画面の色を変えます。

コール手順

A	スクリーンモード
【FORCLR(F3E9H)】	前景色
【BAKCLR(F3EAH)】	背景色
【BDRCLR(F3EBH)】	周辺色

戻り値

なし

変更レジスタ

すべて

CLSSUB（0115H / SUB） MSX2

機　能

画面をクリアします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

DSPFNK（011DH / SUB） MSX2

機　能

ファンクションキーを表示します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

WRTVDP（012DH / SUB） MSX2

機　能

VDPのレジスタにデータを書き込みます。

コール手順

C　　VDPのレジスタ番号
B　　書き込むデータ

戻り値

なし

変更レジスタ

AF、BC

VDPSTA (0131H/SUB)　MSX2

機　能

VDPのレジスタの内容を獲得します。

コール手順

A　　VDPのレジスタ番号（0～9）

戻り値

A　　データ

変更レジスタ

F

SETPAG (013DH/SUB)　MSX2

機　能

VRAMのページを切り換えます。

コール手順

【DDPAGE(FAF5H)】　ディスプレイページ番号
【ACPAGE(FAF6H)】　アクティブページ番号

戻り値

なし

変更レジスタ

AF

INIPLT（0141H / SUB） MSX2

機　能

パレットを初期化します。現在のパレットはVRAMにセーブされています。

コール手順

なし

戻り値

なし

変更レジスタ

AF、BC、DE

RSTPLT（0145H / SUB） MSX2

機　能

パレットをVRAMからリストアします。

コール手順

なし

戻り値

なし

変更レジスタ

AF、BC、DE

GETPLT (0149H / SUB) MSX2

機　能

パレットからカラーコードを獲得します。

コール手順

A　パレット番号（0～15）

戻り値

B　上位4ビットに赤のコード
　下位4ビットに青のコード
C　下位4ビットに緑のコード

変更レジスタ

AF、DE

SETPLT (014DH / SUB) MSX2

機　能

カラーコードをパレットにセットします。

コール手順

D　パレット番号（0～15）
A　上位4ビットに赤のコード
　下位4ビットに青のコード
E　下位4ビットに緑のコード

戻り値

なし

変更レジスタ

AF

BEEP（017DH / SUB） MSX2

機　能

ブザーを鳴らします。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

PROMPT（0181H / SUB） MSX2

機　能

プロンプトを表示します。

コール手順

なし

戻り値

なし

変更レジスタ

すべて

NEWPAD（01ADH / SUB） MSX2

機　能

マウス、ライトペンの状態を調べます。

コール手順

A　以下のデータを入れてコールします。カッコ内は戻り値です(「第6部 5.4 タッチパネル、ライトペン、マウス、トラックボールの使用法」参照)。

8	ライトペンの接続状態を返す (FFHで有効)
9	X座標を返す
10	Y座標を返す
11	ライトペンスイッチの状態を返す (押されたときFFH)
12	マウスのポート1への接続状態を返す (FFHで有効)
13	X方向のオフセットを返す
14	Y方向のオフセットを返す
15	常に0
16	マウスのポート2への接続状態を返す (FFHで有効)
17	X方向のオフセットを返す
18	Y方向のオフセットを返す
19	常に0

戻り値

A

変更レジスタ

すべて

CHGMDP (01B5H / SUB)　MSX2

機　能

VDPのモードを変えます。パレットは初期化します。

コール手順

A　スクリーンモード (0〜8)

戻り値

なし

変更レジスタ

すべて

KNJPRT（01BDH / SUB） MSX2＋

機　能

グラフィック画面（スクリーンモード5〜8）に漢字を表示します。

コール手順

BC　JIS漢字コード（MSX2は第一水準、MSX2＋は第一、第二水準）

A　表示モード

表示モードはBASICのPUT KANJI命令と同様に以下の意味を持つ

0　16×16ドットで表示

1　偶数番目のドットを表示

2　奇数番目のドットを表示

【GRPACX(FCB7H)】　X座標

【GRPACY(FCB9H)】　Y座標

戻り値

なし

変更レジスタ

AF

REDCLK（01F5H / SUB） MSX2

機　能

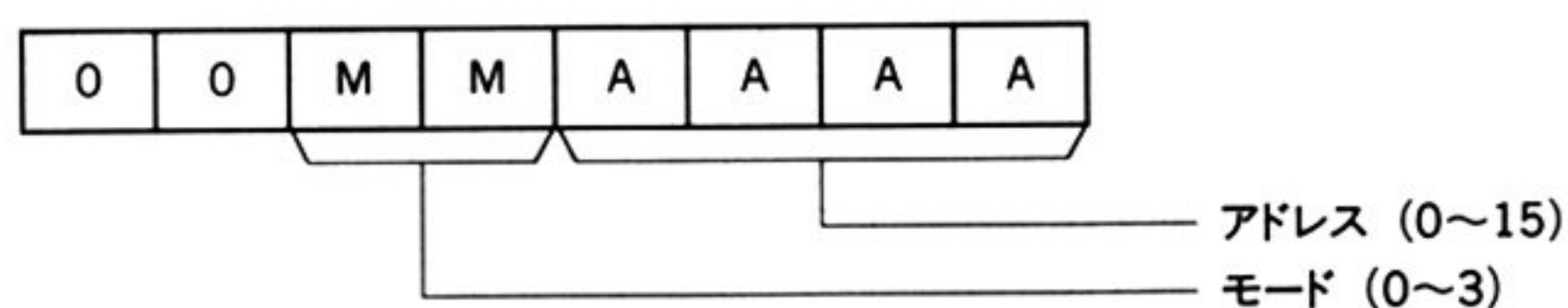
CLOCK-ICのデータを読み出します。

コール手順

C　読み込むCLOCK-ICのレジスタアドレス（「第6部 6章 CLOCKとバッテリバックアップメモリ」参照）

0	0	M	M	A	A	A	A

アドレス（0〜15）

モード（0〜3）

戻り値

A　　読み込んだデータ（下位4ビットのみ有効）

変更レジスタ

F

WRTCLK（01F9H / SUB）　MSX2

機　能

CLOCK-ICにデータを書き込みます。

コール手順

C　　書き込むCLOCK-ICのレジスタアドレス

A　　書き込むデータ

戻り値

なし

変更レジスタ

F

A.2 ワークエリア

■ディスクドライブ用（ディスクがあるときのみ有効）

アドレス	ラベル	長さ	初期値	内　容
F323	DISKVE	2		ディスクエラー処理ルーチンへのポインタへのポインタ
F325	BREAKV	2		CTRL + C 処理ルーチンへのポインタへのポインタ
F341	RAMAD 0	1		ページ0のRAMのスロットアドレス
F342	RAMAD 1	1		ページ1のRAMのスロットアドレス
F343	RAMAD 2	1		ページ2のRAMのスロットアドレス
F344	RAMAD 3	1		ページ3のRAMのスロットアドレス
F348	MASTERS	1		マスターカートリッジのスロットアドレス

■インタースロットコール用サブルーチン

アドレス	ラベル	長さ	初期値	内　容
F380	RDPRIM	5		基本スロットから読み込み
F385	WRPRIM	7		基本スロットへ書き込み
F38C	CLPRIM	14		基本スロットコール

■ USR関数のマシン語プログラムの開始番地、テキスト画面
(MSX2、MSX2＋の場合、テキスト画面用のワークエリアの初期値は変化する)

アドレス	ラベル	長さ	初期値	内　容
F39A	USRTAB	20	475AH	USR関数(0〜9)のマシン語プログラムの開始番地。定義前はエラールーチン（FCERR）を指す
F3AE	LINL40	1	39	SCREEN0のときの1行の幅、WIDTH文で設定
F3AF	LINL32	1	29	SCREEN1のときの1行の幅、WIDTH文で設定
F3B0	LINLEN	1	29	現在の画面の1行の幅
F3B1	CRTCNT	1	24	現在の画面の行数
F3B2	CLMLST	1	14	PRINT文の制御に使用、LINLEN-(LINLEN MOD 14)-14

■ VRAMの各テーブルの番地、その他の画面設定

アドレス	ラベル	長さ	初期値	内　容
F3B3	TXTNAM	2	0000H	SCREEN 0のパターンネームテーブル
F3B5	TXTCOL	2	0800H	SCREEN 0のカラーテーブル(MSX2、MSX2+のみ)
F3B7	TXTCGP	2	0800H	SCREEN 0のパターンジェネレータテーブル
F3B9	TXTATR	2		未使用
F3BB	TXTPAT	2		未使用
F3BD	T32NAM	2	1800H	SCREEN 1のパターンネームテーブル
F3BF	T32COL	2	2000H	SCREEN 1のカラーテーブル
F3C1	T32CGP	2	0000H	SCREEN 1のパターンジェネレータテーブル
F3C3	T32ATR	2	1B00H	SCREEN 1のスプライトアトリビュートテーブル
F3C5	T32PAT	2	3800H	SCREEN 1のスプライトジェネレータテーブル
F3C7	GRPNAM	2	1800H	SCREEN 2のパターンネームテーブル
F3C9	GRPCOL	2	2000H	SCREEN 2カラーテーブル
F3CB	GRPCGP	2	0000H	SCREEN 2パターンジェネレータテーブル
F3CD	GRPATR	2	1B00H	SCREEN 2スプライトアトリビュートテーブル
F3CF	GRPPAT	2	3800H	SCREEN 2スプライトジェネレータテーブル
F3D1	MLTNAM	2	0800H	SCREEN 3のパターンネームテーブル
F3D3	MLTCOL	2		未使用
F3D5	MLTCGP	2	0000H	SCREEN 3のパターンジェネレータテーブル
F3D7	MLTATR	2	1B00H	SCREEN 3のスプライトアトリビュートテーブル
F3D9	MLTPAT	2	3800H	SCREEN 3のスプライトジェネレータテーブル
F3DB	CLIKSW	1	FFH	キークリックスイッチ（0＝OFF、NZ＝ON)
F3DC	CSRY	1		カーソルのY座標
F3DD	CSRX	1		カーソルのX座標
F3DE	CNSDFG	1	FFH	ファンクションキー表示スイッチ(0＝非表示、NZ＝表示)

■ VDPレジスタのセーブエリアなど

アドレス	ラベル	長さ	初期値	内　容
F3DF	RG0SAV	1		VDPレジスタ0の値
F3E0	RG1SAV	1		VDPレジスタ1の値
F3E1	RG2SAV	1		VDPレジスタ2の値
F3E2	RG3SAV	1		VDPレジスタ3の値
F3E3	RG4SAV	1		VDPレジスタ4の値
F3E4	RG5SAV	1		VDPレジスタ5の値
F3E5	RG6SAV	1		VDPレジスタ6の値
F3E6	RG7SAV	1		VDPレジスタ7の値
F3E7	STATFL	1		VDPのSTATUS(MSX2、MSX2+ではSTATUSレジスタ0の値)

アドレス	ラベル	長さ	初期値	内　容
F3E8	TRGFLG	1	FFH	ジョイスティックのトリガボタンの状態
F3E9	FORCLR	1	15	前景色（フォアグラウンドカラー）省略値
F3EA	BAKCLR	1	4	背景色（バックグラウンドカラー）省略値
F3EB	BDRCLR	1	7	周辺色（ボーダーカラー）省略値
F3EC	MAXUPD	3	JMP 0	LINE 文が内部で使用（C3H、00H、00H）
F3EF	MINUPD	3	JMP 0	LINE 文が内部で使用（C3H、00H、00H）
F3F2	ATRBYT	1	15	グラフィック使用時の色、アトリビュートバイト
F3F3	QUEUES	2	F959H	PLAY 文実行時のキューテーブルを指す
F3F5	FRCNEW	1	FFH	インタープリタが内部で使用
F3F6	SCNCNT	1	1	キースキャンの時間間隔
F3F7	REPCNT	1	50	キーのオートリピート開始までの時間間隔
F3F8	PUTPNT	2	FBF0H	キーバッファへの書き込みを行う番地を指す
F3FA	GETPNT	2	FBF0H	キーバッファからの読み込みを行う番地を指す

■カセット用パラメータなど

アドレス	ラベル	長さ	初期値	内　容
F3FC	CS120	10	83	1200 ボーのビット 0 を表す LOW の幅
			92	1200 ボーのビット 0 を表す HIGH の幅
			38	1200 ボーのビット 1 を表す LOW の幅
			45	1200 ボーのビット 1 を表す HIGH の幅
			15	ショートヘッダビットの長さ、HEADLEN（=2000）＊2/256
			37	2400 ボーのビット 0 を表す LOW の幅
			45	2400 ボーのビット 0 を表す HIGH の幅
			14	2400 ボーのビット 1 を表す LOW の幅
			22	2400 ボーのビット 1 を表す HIGH の幅
			31	ショートヘッダビットの長さ、HEADLEN（=2000）＊4/256
F406	LOW	2	83	LOW01、HIGH01、デフォルト 1200 ボー、現在のボーレートの bit0 を表す LOW、HIGH の幅。SCREEN 文で設定
F408	HIGH	2	33	LOW11、HIGH11、デフォルト 1200 ボー、現在のボーレートの bit1 を表す LOW、HIGH の幅。SCREEN 文で設定
F40A	HEADER	1	15	HEADLEN＊2/256、現在のボーレートのショートヘッダ用のヘッダビット（HEADLEN=2000）。SCREEN 文で設定
F40B	ASPCT1	2	1	256/アスペクト比、SCREEN 文で設定し CIRCLE 文で使用
F40D	ASPCT2	2	1	256＊アスペクト比、SCREEN 文で設定し CIRCLE 文で使用
F40F	ENDPRG	5	":"	RESUME NEXT文のための仮のプログラムの終り

■ BASIC が内部で使うワーク

アドレス	ラベル	長さ	初期値	内　容
F414	ERRFLG	1	0	エラーコードの保存
F415	LPTPOS	1	0	プリンタのヘッド位置
F416	PRTFLG	1	0	プリンタへ出力するかどうか
F417	NTMSXP	1	0	MSX 用プリンタなら 0
F418	RAWPRT	1	0	NZ なら RAW MODE（文字コード変換なし）でプリンタ出力
F419	VLZADR	2	0	VAL 関数で置き換えられる文字のアドレス
F41B	VLZDAT	1	0	VAL 関数で 0 に置き換わる文字
F41C	CURLIN	2	FFFFH	BASIC が現在実行中の行番号
F41F	KBUF	318		クランチバッファ、BUF から中間言語に変換されて入る
F55D	BUFMIN	1	","	INPUT 文用のコンマ
F55E	BUF	258		タイプした文字がアスキーコードで入るバッファ
F660	ENDBUF	1	0	BUF の内容がオーバーフローするのを防ぐ
F661	TTYPOS	1	0	BASIC が内部で持つ仮想的なカーソル位置
F662	DIMFLG	1	0	BASIC が配列変数を単純変数と区別するためのフラグ
F663	VALTYP	1	0	変数の型の識別に使用
F664	OPRTYP	1	0	演算子保存、またはクランチできる語かどうかのフラグ
F665	DONUM	1	0	クランチ用のフラグ
F666	CONTXT	2	0	CHGET で使うテキストポインタの保存
F668	CONSAV	1	0	CHGET が呼ばれた後で定数の内部形（トークン）を保存
F669	CONTYP	1	0	保存した定数のタイプ
F66A	CONLO	8	0	保存した定数の値
F672	MEMSIZ	2		BASIC が使用するメモリの最上位番地
F674	STKTOP	2		スタックとして使える最上位番地、CLEAR 文で変更される
F676	TXTTAB	2		BASIC のテキストエリアの先頭番地
F678	TEMPPT	2	F67AH	テンポラリディスクリプタの空きエリアの先頭番地
F67A	TEMPST	30		3 * NUMTMP 分使用される
F698	DSCTMP	3		文字列関数の答えへのストリングディスクリプタ
F69B	FRETOP	2		文字列領域の空きエリアの先頭番地
F69D	TEMP3	2	0	ガベージコレクション用
F69F	TEMP8	2	0	ガベージコレクション用
F6A1	ENDFOR	2	0	FOR 文の次の番地
F6A3	DATLIN	2	0	READ 文で読まれた DATA 文の行番号
F6A5	SUBFLG	1	0	USR 関数などで配列を使うときのフラグ
F6A6	FLGINP	1	0	INPUT 文や READ 文で使われるフラグ
F6A7	TEMP	2		ステートメントコード用の一時保存場所

アドレス	ラベル	長さ	初期値	内　容
F6A9	PTRFLG	1	0	変換する行番号がなければ0、あれば0以外
F6AA	AUTFLG	1	0	AUTOコマンドが無効なら0、有効なら0以外
F6AB	AUTLIN	2	0	AUTOで入力された最新の行番号
F6AD	AUTINC	2	0	AUTOコマンドの行番号の増分
F6AF	SAVTXT	2		RESUME文で復帰するためのテキストのアドレス
F6B1	SAVSTK	2		エラーが起きたときの回復ルーチン用スタックの保存
F6B3	ERRLIN	2	0	エラーが起きたときの行番号
F6B5	DOT	2	0	現在行、LIST.などで使う
F6B7	ERRTXT	2	0	エラーが起きた行番号、RESUME文用
F6B9	ONELIN	2	0	エラーが起きたときの飛び先の行
F6BB	ONEFLG	1	0	エラーによる割り込みルーチン実行中は1、他の場合は0
F6BC	TEMP2	2	0	一時保存用
F6BE	OLDLIN	2	0	CTRL＋STOP、END、STOPで設定される旧行番号
F6C0	OLDTXT	2	0	次に実行する文のテキストアドレス
F6C2	VARTAB	2		単純変数の開始番地、NEW文で(TXTTAB)+2に設定される
F6C4	ARYTAB	2		配列テーブルの開始番地
F6C6	STREND	2		BASICがテキストや変数に使用中のメモリの最後の番地
F6C8	DATPTR	2	0	READ文実行で読まれたデータのテキストアドレス
F6CA	DEFTBL	26	8	各英文字で始まる変数のデフォルトの型を保持する

■ユーザー関数のパラメータ用ワーク

アドレス	ラベル	長さ	初期値	内　容
F6E4	PRMSTK	2	0	ガベージコレクション用スタック上の以前の定義ブロック
F6E6	PRMLEN	2	0	処理対象のテーブルのバイト数
F6E8	PARM1	100	0	PRMSIZで設定される処理パラメータ定義テーブル
F74C	PRMPRV	2	PRMSTK	以前のパラメータブロックのポインタ、ガベージコレクション用
F74E	PRMLN2	2	0	パラメータブロックの大きさ
F750	PARM2	100	0	PRMSIZで設定されるパラメータの保存場所
F7B4	PRMFLG	1	0	PARM1がサーチ済みかどうかのフラグ
F7B5	ARYTA2	2	0	サーチの終点
F7B7	NOFUNS	1	0	処理対象関数がない場合は0
F7B8	TEMP9	2	0	ガベージコレクション用

アドレス	ラベル	長さ	初期値	内　容
F7BA	FUNACT	2	0	処理対象関数の数
F7BC	SWPTMP	8	0	SWAP 文の最初の変数の値の一時保存場所
F7C4	TRCFLG	1	0	トレースフラグ、0 = TRACE OFF、NZ = TRACE ON

■ Math-pack 用ワークエリア

アドレス	ラベル	長さ	初期値	内　容
F7C5	FBUFFR	43		Math-pack ルーチンが一時保存場所として使用する
F7F0	DECTMP	2		10 進数を浮動小数点数にするときに使用する
F7F2	DECTM2	2		除算ルーチンで使用する
F7F4	DECCNT	1		除算ルーチンで使用する
F7F6	DAC	16		演算の対象となる値を設定するエリア
F806	HOLD8	48		10 進数の乗算のためのレジスタ保存エリア
F836	HOLD2	8		Math-Pack ルーチンが内部で使用する
F83E	HOLD	8		Math-Pack ルーチンが内部で使用する
F847	ARG	16		DAC との演算対象となる値を設定するエリア
F857	RNDX	8		最初の乱数を倍精度実数で設定するエリア

■ BASIC インタープリタが使うデータエリア

アドレス	ラベル	長さ	初期値	内　容
F85F	MAXFIL	1	1	MAXFILES 文で設定されるファイル番号の最大値
F860	FILTAB	2		ファイルデータの先頭番地
F862	NULBUF	2		SAVE、LOAD で BASIC インタープリタが使うバッファ
F864	PTRFIL	2	0	指定ファイルのファイルデータのある位置
F866	RUNFLG	0	0	ロード後実行なら NZ
F866	FILNAM	11	" "	ファイル名の保存エリア
F871	FILNM2	11	" "	ファイル名の保存エリア
F87C	NLONLY	1	0	プログラムロード中は NZ
F87D	SAVEND	2	0	セーブするマシン語プログラムの最終番地
F87F	FNKSTR	160		ファンクションキーの文字列保存エリア、16×10
F91F	CGPNT	3		ROM 上の文字フォント格納スロットアドレスとアドレス
F922	NAMBAS	2		現在のパターンネームテーブルのベース番地
F924	CGPBAS	2		現在のパターンジェネレータテーブルのベース番地
F926	PATBAS	2		現在のスプライトジェネレータテーブルのベース番地

アドレス	ラベル	長さ	初期値	内　容
F928	ATRBAS	2		現在のスプライトアトリビュートテーブルのベース番地
F92A	CLOC	2		グラフィックルーチンが内部で使用する
F92C	CMASK	1		グラフィックルーチンが内部で使用する
F92D	MINDEL	2		グラフィックルーチンが内部で使用する
F92F	MAXDEL	2		グラフィックルーチンが内部で使用する

■ CIRCLE、PAINT 文で使うデータエリア

アドレス	ラベル	長さ	初期値	内　容
F931	ASPECT	2		円の縦横比率。CIRCLE 文の＜比率＞により設定される
F933	CENCNT	2		CIRCLE 文が内部で使用する
F935	CLINEF	1		中心へ線を引くかどうかのフラグ
F936	CNPNTS	2		プロットする点
F938	CPLOTF	1		CIRCLE 文が内部で使用する
F939	CPCNT	2		円の 1/8 分割の数
F93B	CPCNT8	2		CIRCLE 文が内部で使用する
F93D	CRCSUM	2		CIRCLE 文が内部で使用する
F93F	CSTCNT	2		CIRCLE 文が内部で使用する
F941	CSCLXY	1		X と Y のスケール
F942	CSAVEA	2		ADVGRP の保存エリア
F944	CSAVEM	1		ADVGRP の保存エリア
F945	CXOFF	2		中心からの X のオフセット
F947	CYOFF	2		中心からの Y のオフセット
F949	LOHMSK	1		PAINT 文が内部で使用する
F94A	LOHDIR	1		PAINT 文が内部で使用する
F94B	LOHADR	2		PAINT 文が内部で使用する
F94D	LOHCNT	2		PAINT 文が内部で使用する
F94F	SKPCNT	2		スキップカウント
F951	MOVCNT	2		移動カウント
F953	PDIREC	1		ペイント方向
F954	LFPROG	1		PAINT 文が内部で使用する
F955	RTPROG	1		PAINT 文が内部で使用する

■ PLAY 文で使用するデータエリア

アドレス	ラベル	長さ	初期値	内容
F956	MCLTAB	2		PLAY 文マクロ、DRAW 文マクロのテーブルの先頭を指す
F958	MCLFLG	1		PLAY・DRAW の指示
F959	QUETAB	24		キューテーブル、4 キュー×6 バイト
F971	QUEBAK	4		BCKQ で使用
(内部定義 MUSQLN、RSIQLN)				
F975	VOICAQ	n		n は MUSQLN、音声 1 のキュー
F9F5	VOICBQ	n		n は MUSQLN、音声 2 のキュー
FA75	VOICCQ	n		n は MUSQLN、音声 3 のキュー

■ MSX2 で追加されたワークエリア

アドレス	ラベル	長さ	初期値	内容
FAF5	DPPAGE	1		ディスプレイページ番号
FAF6	ACPAGE	1		アクティブページ番号
FAF7	AVCSAV	1		AV コントロールポートの保存
FAF8	EXBRSA	1		SUB ROM のスロットアドレス
FAF9	CHRCNT	1		ローマ字カナ変換で使用、キャラクタカウンタで 0〜2
FAFA	ROMA	2		ローマ字カナ変換で使用、バッファ中のキャラクタ
FAFC	MODE	1		ローマ字カナ変換のモード、VRAM サイズなど
各ビットの意味	MSB	7		0 =ひらがな (ローマ字変換)、1 =カタカナ
		6		0 =第 2 水準漢字 ROM なし、1 =第 2 水準漢字 ROM あり
		5		0 = SCREEN 10、1 = SCREEN 11 (MSX2+で、RGB 処理のフラグ)
		4		0 =クリッピングする、1 =クリッピングしない
		3		0 =マスクしない、1 =マスクする (SCREEN 0〜3 の VRAM アドレス)
		2		0 } 16K VRAM 0 / 0 } 64K VRAM 1 / 1 } 128K VRAM 0 (bit2 と bit1 のセットで VRAM サイズがわかる)
		1		
	LSB	0		0 =変換しない (ローマ字変換)、1 =変換する

注 意　bit3 は、MSX2 以降で SCREEN 0〜3 の VRAM アドレスを指定する際に、3FFFH と AND を取って設定するかどうかのフラグ。SCREEN 4 以降では常にマスクしない。

アドレス	ラベル	長さ	初期値	内容
FAFD	NORUSE	1		漢字ドライバが使用
FAFE	XSAVE	2		I0000000 XXXXXXXX
FB00	YSAVE	2		×0000000 YYYYYYYY I =1 ライトペンのインタラプト要求あり

アドレス	ラベル	長さ	初期値	内　容
				OOOOOOO =符号無しオフセット
				XXXXXXXX= X 座標
				YYYYYYYY= Y 座標
FB02	LOGOPR	1		V9938 以降のロジカルオペレーションコード

■ RS-232C で使うデータエリア

アドレス	ラベル	長さ	初期値	内　容
FB03	RSTMP	50		RS-232C またはディスクが使用する
FB03	TOCNT	1		RS-232C が内部で使用する
FB04	RSFCB	1		RS-232C の LOW アドレス
		1		RS-232C の HIGH アドレス
FB06	RSIQLN	1		RS-232C が内部で使用する
FB07	MEXBIH	5		FB07H+0　RST 30H(0F7H)
				FB07H+1　バイトデータ
				FB07H+2　(LOW)
				FB07H+3　(HIGH)
				FB07H+4　RET(0C9H)
FB0C	OLDSTT	5		FB0CH+0　RST 30H(0F7H)
				FB0CH+1　バイトデータ
				FB0CH+2　(LOW)
				FB0CH+3　(HIGH)
				FB0CH+4　RET(0C9H)
FB11	OLDINT	5		FB11H+0　RST 30H(0F7H)
				FB11H+1　バイトデータ
				FB11H+2　(LOW)
				FB11H+3　(HIGH)
				FB11H+4　RET(0C9H)
FB16	DEVNUM	1		RS-232C が内部で使用する
FB17	DATCNT	3		FB17H+0　バイトデータ
				FB17H+1　バイトポインタ
				FB17H+2　バイトポインタ
FB1A	ERRORS	1		RS-232C が内部で使用する
FB1B	FLAGS	1		RS-232C が内部で使用する
FB1C	ESTBLS	1		RS-232C が内部で使用する
FB1D	COMMSK	1		RS-232C が内部で使用する
FB1E	LSTCOM	1		RS-232C が内部で使用する
FB1F	LSTMOD	1		RS-232C が内部で使用する
FB20	HOKVLD	1		拡張 BIOS の有無
FB21	DRVTBL	8		DISK ROM のスロットアドレスなど

■ PLAY 文で使用するデータエリア（以下は MSX1 と共通）

アドレス	ラベル	長さ	初期値	内　容
FB35	PRSCNT	1		
FB36	SAVSP	2		PLAY 文実行中にスタックポインタを保存
FB38	VOICEN	1		解釈中の現在の音声
FB39	SAVVOL	2		休止のための音量保存
FB3B	MCLLEN	1		PLAY 文が内部で使用する
FB3C	MCLPTR	2		PLAY 文が内部で使用する
FB3E	QUEUEN	1		PLAY 文が内部で使用する
FB3F	MUSICF	1		音楽演奏用の割り込みフラグ
FB40	PLYCNT	1		キューされている PLAY 文の数
（音声スタティックデータエリアから＋0～＋32?）				
FB41	VCBA	n		n は VCBSIZ、音声 1 のスタティックデータ
FB66	VCBB	n		n は VCBSIZ、音声 2 のスタティックデータ
FB8B	VCBC	n		n は VCBSIZ、音声 3 のスタティックデータ

■データエリア

アドレス	ラベル	長さ	初期値	内　容
FBB0	ENSTOP	1		0 以外は「CTRL ＋ SHIFT ＋ GRAPH ＋ カナ」でウォームスタート可能
FBB1	BASROM	1		0 以外は BASIC のテキストが ROM にある
FBB2	LINTTB	24		ラインターミネーターテーブル
FBCA	FSTPOS	2		INLIN で入力した行の最初の文字の位置
FBCC	CODSAV	1		カーソルのためのコード保存場所
FBCD	FNKSWI	1		どのファンクションキーが表示されているか
FBCE	FNKFLG	10		割り込み処理対象デバイスになっている F キーを示す
FBD8	ONGSBF	1		広域イベントフラグ
FBD9	CLIKFL	1		キークリックフラグ
FBDA	OLDKEY	11		旧キーの状態
FBE5	NEWKEY	11		新キーの状態
（内部定義 SFTKEY、NEWKEY＋6、RETURN、CTRL、SHIFT キーの状態）				
FBF0	KEYBUF	40		キーコードバッファ
FC18	BUFEND	0		KEYBUF の終り
FC18	LINWRK	40		スクリーンハンドラ用一時保存場所
FC40	PATWRK	8		パターンコンバータ用一時保存場所
FC48	BOTTOM	2		実装 RAM の先頭（低位）番地
FC4A	HIMEM	2		利用可能なメモリーの最上位番地
FC4C	TRPTBL	n		n は 3 * NUMTPR。割り込み処理で使うトラップテーブル

アドレス	ラベル	長さ	初期値	内　容
FC9A	RTYCNT	1		BASIC が内部で使用する
FC9B	INTFLG	1		CTRL ＋ STOP が押されたときなど、ここに03H を入れることによりストップする
FC9C	PADY	1		パドルの X 座標
FC9D	PADX	1		パドルの Y 座標
FC9E	JIFFY	2		PLAY 文が内部で使用する
FCA0	INTVAL	2		インターバルの間隔。ON INTERVAL GOSUB 文により設定される
FCA2	INTCNT	2		インターバルのためのカウンタ
FCA4	LOWLIM	1		カセットテープから読み込み中に使う
FCA5	WINWID	1		カセットテープから読み込み中に使う
FCA6	GRPHED	1		グラフィック文字を出すときのフラグ
FCA7	ESCCNT	1		エスケープシーケンスのカウンタ
FCA8	INSFLG	1		挿入モードのフラグ
FCA9	CSRSW	1		カーソル表示の有無
FCAA	CSTYLE	1		カーソルの形
FCAB	CAPST	1		CAPS キーの状態
FCAC	KANAST	1		かな キーの状態
FCAD	KANAMD	1		かな文字が JIS 配列なら 0 でない値
FCAE	FLBMEM	1		BASIC プログラムをロード中は 0
FCAF	SCRMOD	1		スクリーンモードの番号
FCB0	OLDSCR	1		スクリーンモード保存場所
FCB1	CASPRV	1		CAS:が使う文字保存場所
FCB2	BRDATR	1		PAINT のボーダーカラー
FCB3	GXPOS	2		X 座標
FCB5	GYPOS	2		Y 座標
FCB7	GRPACX	2		グラフィックアキュムレータ（X 座標）
FCB9	GRPACY	2		グラフィックアキュムレータ（Y 座標）
FCBB	DRWFLG	1		DRAW 文で使用するフラグ
FCBC	DRWSCL	1		DRAW のスケーリングファクタ。0 ならスケーリングなし
FCBD	DRWANG	1		DRAW の角度。0〜3
FCBE	RUNBNF	1		BLOAD か BSAVE かどちらでもないか
FCBF	SAVENT	2		BSAVE の開始番地
FCC1	EXPTBL	4		各スロットの拡張の有無を示すフラグテーブル
FCC5	SLTTBL	4		各拡張スロットレジスタ用の現在のスロット選択状況
FCC9	SLTATR	64		各スロット用に属性を保存
FD09	SLTWRK	128		各スロット用に特定のワークエリアを確保
FD89	PROCNM	16		CALL 文による拡張文の名前。0 は終り
FD99	DEVICE	1		カートリッジ用の装置識別に使う

■割り込み処理、他のコンソール入出力装置使用、文字セットやキー配列の変更

アドレス	ラベル	長さ	初期値	内　容
FD9A	H.KEYI	5		割り込み処理の始め、RS-232C など
FD9F	H.TIMI	5		タイマー割り込み処理の追加
FDA4	H.CHPU	5		CHPUT（1文字出力）の始め
FDA9	H.DSPC	5		DSPCSR（カーソル表示）の始め
FDAE	H.ERAC	5		ERACSR（カーソル消去）の始め
FDB3	H.DSPF	5		DSPFNK（ファンクションキー表示）の始め
FDB8	H.ERAF	5		ERAFNK（ファンクションキー消去）の始め
FDBD	H.TOTE	5		TOTEXT（画面をテキストモードにする）の始め
FDC2	H.CHGE	5		CHGET（1文字取り出し）の始め
FDC7	H.INIP	5		INIPAT（文字パターンの初期化）の始め
FDCC	H.KEYC	5		KEYCOD（キーコード変換）の始め
FDD1	H.KEYA	5		KYEASY（KeY EASY）の始め
FDD6	H.NMI	5		NMI（ノンマスカブルインタラプト）の始め
FDDB	H.PINL	5		PINLIN（プログラム行入力）の始め
FDE0	H.QINL	5		QINLIN（「?」を表示して行入力）の始め
FDE5	H.INLI	5		INLIN（行入力）の始め
FDEA	H.ONGO	5		ONGOTP（ON GOTO）の始め

■ディスク装置接続

アドレス	ラベル	長さ	初期値	内　容
FDEF	H.DSKO	5		DSKO$（ディスク出力）の始め
FDF4	H.SETS	5		SETS$（セットアトリビュート）の始め
FDF9	H.NAME	5		NAME（リネーム）の始め
FDFE	H.KILL	5		KILL（ファイル削除）の始め
FE03	H.IPL	5		IPL（初期プログラムロード）の始め
FE08	H.COPY	5		COPY（ファイルのコピー）の始め
FE0D	H.CMD	5		CMD（コマンド）の始め
FE12	H.DSKF	5		DSKF（ディスクの空き）の始め
FE17	H.DSKI	5		DSKI（ディスク入力）の始め
FE1C	H.ATTR	5		ATTR$（アトリビュート）の始め
FE21	H.LSET	5		LSET（左詰め代入）の始め
FE26	H.RSET	5		RSET（右詰め代入）の始め
FE2B	H.FIEL	5		FIELD（フィールド）の始め
FE30	H.MKI$	5		MKI$（整数作成）の始め
FE35	H.MKS$	5		MKS$（単精度作成）の始め
FE3A	H.MKD$	5		MKD$（倍精度作成）の始め
FE3F	H.CVI	5		CVI（整数変換）の始め
FE44	H.CVS	5		CVS（単精度変換）の始め

アドレス	ラベル	長さ	初期値	内容
FE49	H.CVD	5		CVD（倍精度変換）の始め
FE4E	H.GETP	5		GETPTR（ファイルポインター取り出し）
FE53	H.SETF	5		SETFIL（ファイルポインター設定）
FE58	H.NOFO	5		NOFOR（OPEN 文に FOR がない）
FE5D	H.NULO	5		NULOPN（空ファイルオープン）
FE62	H.NTFL	5		NTFL0（ファイル番号が 0 でない）
FE67	H.MERG	5		MERGE（プログラムファイルのマージ）
FE6C	H.SAVE	5		SAVE（セーブ）
FE71	H.BINS	5		BINSAV（機械語セーブ）
FE76	H.BINL	5		BINLOD（機械語ロード）
FE7B	H.FILE	5		FILES（ファイル一覧表示）
FE80	H.DGET	5		DGET（ディスク GET）
FE85	H.FILO	5		FILOU1（ファイル出力）
FE8A	H.INDS	5		INDSKC（ディスクの属性を入力）
FE8F	H.RSLF	5		（前のドライブを再び選択する）
FE94	H.SAVD	5		（現在選択しているドライブを保存する）
FE99	H.LOC	5		（LOC 関数、場所を示す）
FE9E	H.LOF	5		（LOF 関数、ファイルの長さ）
FEA3	H.EOF	5		（EOF 関数、ファイルの終り）
FEA8	H.FPOS	5		（FPOS 関数、ファイルの場所）
FEAD	H.BAKU	5		BAKUPT（バックアップ）

■論理装置名の拡張

アドレス	ラベル	長さ	初期値	内容
FEB2	H.PARD	5		PARDEV（装置名の取り出し）
FEB7	H.NODE	5		NODEVN（装置名なし）
FEBC	H.POSD	5		POSDSK（ディスク装置）
FEC1	H.DEVN	5		DEVNAM（装置名の処理）
FEC6	H.GEND	5		GENDSP（装置割り当て）

■ BASIC 内部で使用

アドレス	ラベル	長さ	初期値	内容
FECB	H.RUNC	5		RUNC（RUN のためのクリア）
FED0	H.CLEA	5		CLEARC（CLEAR のためのクリア）
FED5	H.LOPD	5		LOPDFT（繰り返しと省略値設定）
FEDA	H.STKE	5		STKERR（スタックエラー）
FEDF	H.ISFL	5		ISFLIO（ファイルの入出力かどうか）

アドレス	ラベル	長さ	初期値	内　容
FEE4	H.OUTD	5		OUTDO（OUT を実行）
FEE9	H.CRDO	5		CRDO（CRLF を実行）
FEEE	H.DSKC	5		DSKCHI（ディスクの属性を入力）
FEF3	H.DOGR	5		DOGRPH（グラフィックを実行）
FEF8	H.PRGE	5		PRGEND（プログラム終了）
FEFD	H.ERRP	5		ERRPRT（エラー表示）
FF02	H.ERRF	5		
FF07	H.READ	5		READY
FF0C	H.MAIN	5		MAIN
FF11	H.DIRD	5		DIRDO（ダイレクトステートメント実行）
FF16	H.FINI	5		
FF1B	H.FINE	5		
FF20	H.CRUN	5		
FF25	H.CRUS	5		
FF2A	H.ISRE	5		
FF2F	H.NTFN	5		
FF34	H.NOTR	5		
FF39	H.SNGF	5		
FF3E	H.NEWS	5		
FF43	H.GONE	5		
FF48	H.CHRG	5		
FF4D	H.RETU	5		
FF52	H.PRTF	5		
FF57	H.COMP	5		
FF5C	H.FINP	5		
FF61	H.TRMN	5		
FF66	H.FRME	5		
FF6B	H.NTPL	5		
FF70	H.EVAL	5		
FF75	H.ONKO	5		
FF7A	H.FING	5		
FF7F	H.ISMI	5		ISMID$（MID$かどうか）
FF84	H.WIDT	5		WIDTHS（WIDTH）
FF89	H.LIST	5		LIST
FF8E	H.BUFL	5		BUFLIN（バッファーライン）
FF93	H.FRQI	5		FRQINT
FF98	H.SCNE	5		
FF9D	H.FRET	5		FRETMP
FFA2	H.PTRG	5		PTRGET（省略値以外の変数使用のポインタ）
FFA7	H.PHYD	5		PHYDIO（物理ディスク入出力）
FFAC	H.FORM	5		FORMAT（ディスクのフォーマット）
FFB1	H.ERRO	5		ERROR（アプリケーションのエラーを処理）

アドレス	ラベル	長さ	初期値	内　容
FFB6	H.LPTO	5		LPTOUT（省略値以外のプリンタで出力）
FFBB	H.LPTS	5		LPTSTT（省略値以外のプリンタで状態を知る）
FFC0	H.SCRE	5		SCREEN
FFC5	H.PLAY	5		PLAY

■拡張 BIOS が使用

アドレス	ラベル	長さ	初期値	内　容
FFCA	FCALL	5		拡張 BIOS が使用
FFCF	DISINT	5		DOS が使用
FFD4	ENAINT	5		DOS が使用

■データエリア

アドレス	ラベル	長さ	初期値	内　容
FFE7	RG8SAV	1		VDP レジスタ#8 のセーブエリア
FFE8	RG9SAV	1		VDP レジスタ#9 のセーブエリア
FFE9	RG10SA	1		VDP レジスタ#10 のセーブエリア
FFEA	RG11SA	1		VDP レジスタ#11 のセーブエリア
FFEB	RG12SA	1		VDP レジスタ#12 のセーブエリア
FFEC	RG13SA	1		VDP レジスタ#13 のセーブエリア
FFED	RG14SA	1		VDP レジスタ#14 のセーブエリア
FFEE	RG15SA	1		VDP レジスタ#15 のセーブエリア
FFEF	RG16SA	1		VDP レジスタ#16 のセーブエリア
FFF0	RG17SA	1		VDP レジスタ#17 のセーブエリア
FFF1	RG18SA	1		VDP レジスタ#18 のセーブエリア
FFF2	RG19SA	1		VDP レジスタ#19 のセーブエリア
FFF3	RG20SA	1		VDP レジスタ#20 のセーブエリア
FFF4	RG21SA	1		VDP レジスタ#21 のセーブエリア
FFF5	RG22SA	1		VDP レジスタ#22 のセーブエリア
FFF6	RG23SA	1		VDP レジスタ#23 のセーブエリア
FFF7		3		システム予約
FFFA	RG25SA	1		VDP レジスタ#25 のセーブエリア
FFFB	RG26SA	1		VDP レジスタ#26 のセーブエリア
FFFC	RG27SA	1		VDP レジスタ#27 のセーブエリア
FFFD		2		システム予約
FFFF		1		拡張スロット選択レジスタ

A.3 VRAMマップ

■SCREEN 0 (WIDTH 40)

TEXT 1

0 4000 8000 C000 10000〜

0000 0200 0400 0600 0800 0A00 0C00 0E00 1000 1200 1400 1600 1800 1A00 1C00 1E00 2000 2200 2400 2600 2800 2A00 2C00 2E00 3000 3200 3400 3600 3800 3A00 3C00 3E00

5000 9000 D000 6000 A000 E000 7000 B000 F000

パターンネームテーブル
パレットテーブル
パターンジェネレータテーブル
3ビット 3ビット R B G P0 R B G P1
パレットテーブル

パターンネームテーブル	0000H〜03BFH	960 bytes
パレットテーブル	0400H〜041FH	32 bytes
パターンジェネレータテーブル	0800H〜0FFFH	2048 bytes

図A.1 SCREEN 0 (WIDTH 40) VRAMマップ

■SCREEN 0 (WIDTH 80)

TEXT 2

パターンネームテーブル	0000H～077FH	1920 bytes
カラーテーブル（ブリンクテーブル）	0800H～090DH	270 bytes
パレットテーブル	0F00H～0F1FH	32 bytes
パターンジェネレータテーブル	1000H～17FFH	2048 bytes

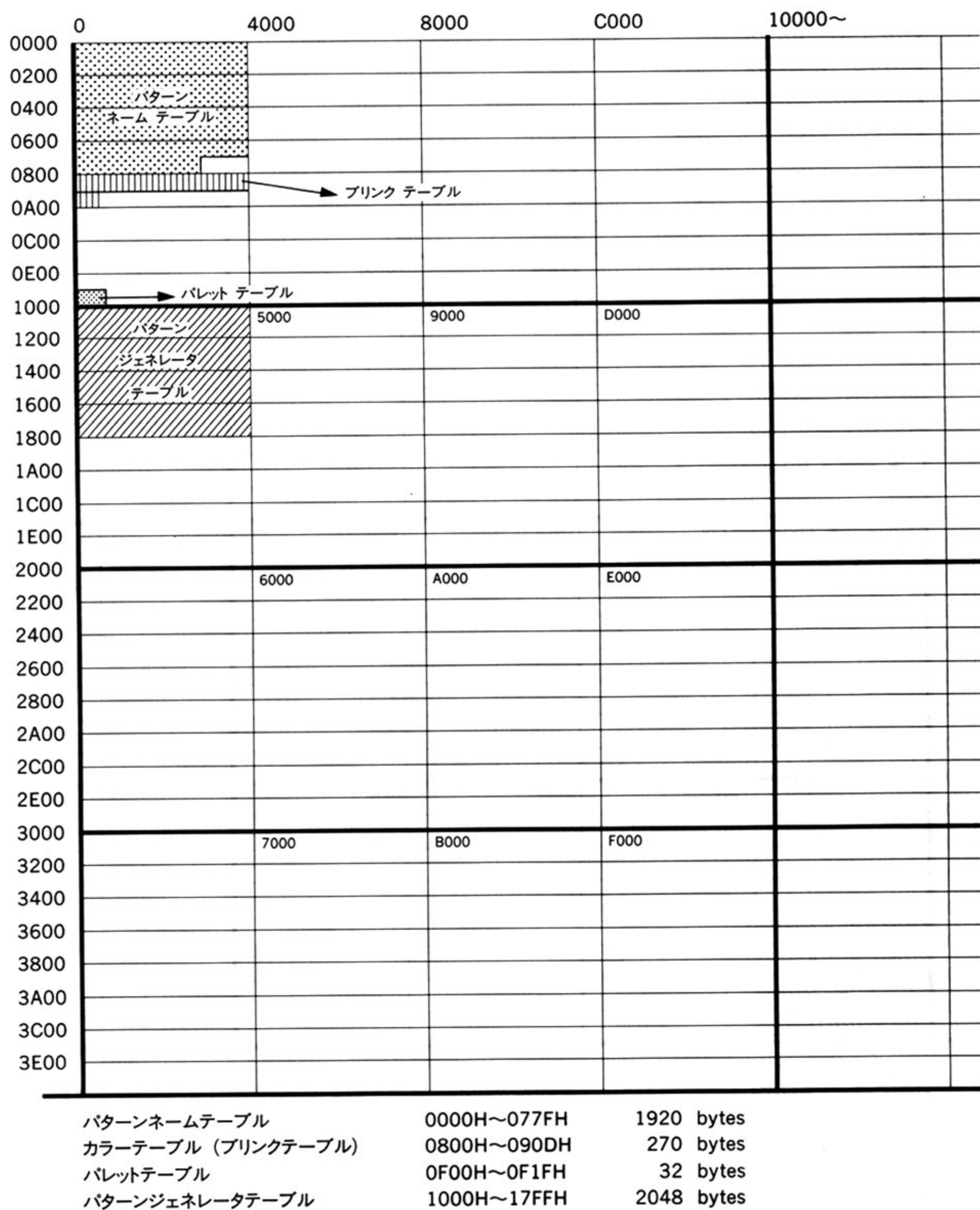



図 A.2 SCREEN 0 (WIDTH 80) VRAM マップ

■SCREEN 1

GRAPHIC 1

0 4000 8000 C000 10000～

0000 0200 0400 0600 0800 0A00 0C00 0E00

パターン
ジェネレータ
テーブル

1000 1200 1400 1600 1800 1A00 1C00 1E00

5000 9000 D000

パターン
ネーム テーブル

スプライト アトリビュート テーブル

2000 2200 2400 2600 2800 2A00 2C00 2E00

A000 E000

パレット テーブル

カラー テーブル

3000 3200 3400 3600 3800 3A00 3C00 3E00

7000 B000 F000

スプライト
ジェネレータ
テーブル

パターンジェネレータテーブル	0000H～07FFH	2048 bytes
パターンネームテーブル	1800H～1AFFH	768 bytes
スプライトアトリビュートテーブル	1B00H～1B7FH	128 bytes
カラーテーブル	2000H～201FH	32 bytes
パレットテーブル	2020H～203FH	32 bytes
スプライトジェネレータテーブル	3800H～3FFFH	2048 bytes

図 A.3 SCREEN 1 VRAM マップ

■SCREEN 2

GRAPHIC 2

パターンジェネレータテーブル	0000H〜17FFH	6144 bytes
パターンネームテーブル	1800H〜1AFFH	768 bytes
スプライトアトリビュートテーブル	1B00H〜1B7FH	128 bytes
パレットテーブル	1B80H〜1B9FH	32 bytes
カラーテーブル	2000H〜37FFH	6144 bytes
スプライトジェネレータテーブル	3800H〜3FFFH	2048 bytes

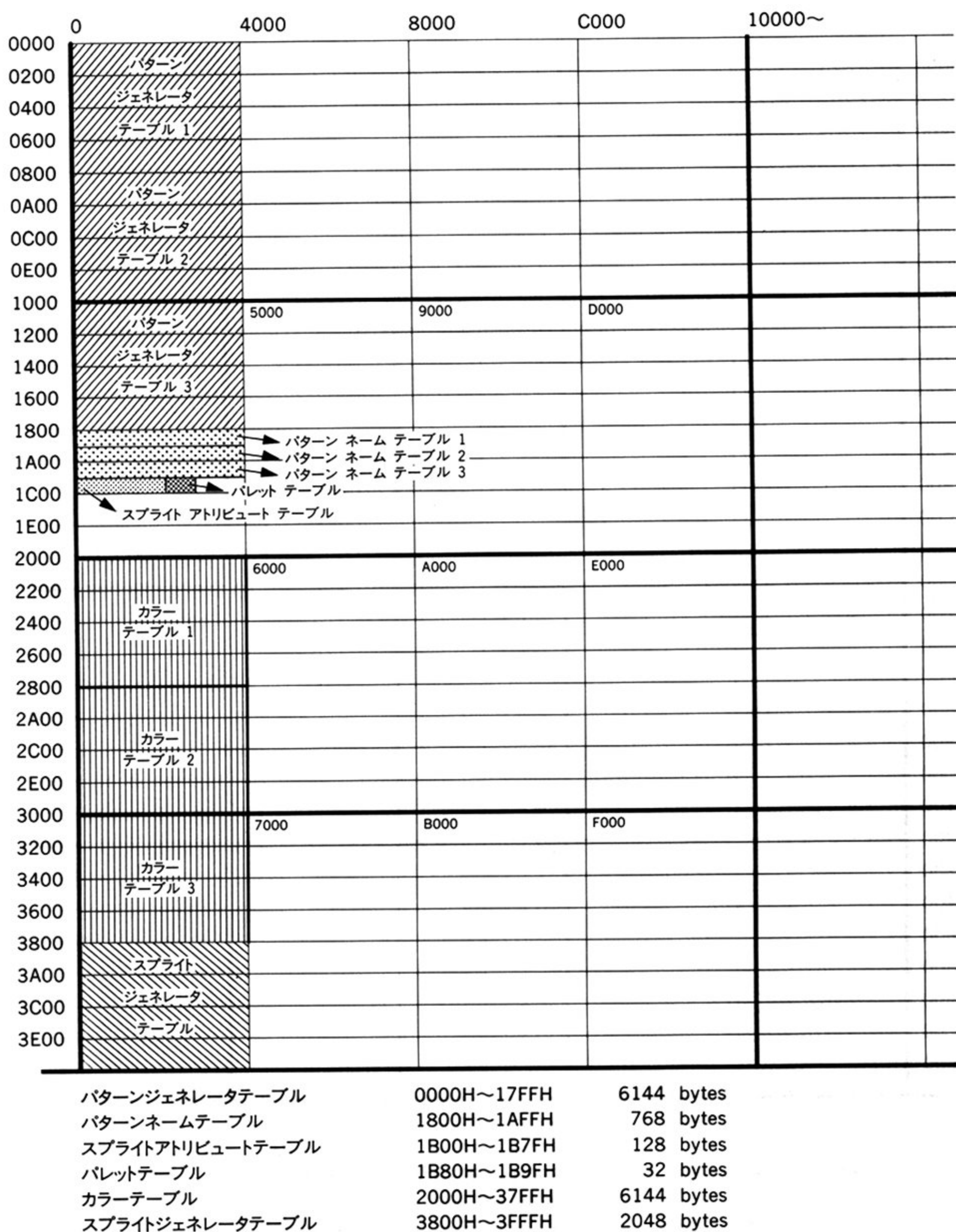



図 A.4 SCREEN 2 VRAM マップ

■SCREEN 3

MULTI COLOR

パターンジェネレータテーブル	0000H〜07FFH	2048 bytes
パターンネームテーブル	0800H〜0AFFH	768 bytes
スプライトアトリビュートテーブル	1B00H〜1B7FH	128 bytes
パレットテーブル	2020H〜203FH	32 bytes
スプライトジェネレータテーブル	3800H〜3FFFH	2048 bytes

図 A.5 SCREEN 3 VRAM マップ

■SCREEN 4

GRAPHIC 3

パターンジェネレータテーブル1
パターンジェネレータテーブル2
パターンジェネレータテーブル3
パターン ネーム テーブル 1
パターン ネーム テーブル 2
パターン ネーム テーブル 3
パレット テーブル
スプライト カラー テーブル
スプライト アトリビュート テーブル
カラーテーブル1
カラーテーブル2
カラーテーブル3
スプライトジェネレータテーブル

パターンジェネレータテーブル	0000H～17FFH	6144 bytes
パターンネームテーブル	1800H～1AFFH	768 bytes
パレットテーブル	1B80H～1B9FH	32 bytes
スプライトカラーテーブル	1C00H～1DFFH	512 bytes
スプライトアトリビュートテーブル	1E00H～1E7FH	128 bytes
カラーテーブル	2000H～37FFH	6144 bytes
スプライトジェネレータテーブル	3800H～3FFFH	2048 bytes

図 A.6 SCREEN 4 VRAM マップ

■SCREEN 5,6

GRAPHIC 4,5

0 4000 8000 C000 10000〜

0000 0200 0400 0600 0800 0A00 0C00 0E00 1000 1200 1400 1600 1800 1A00 1C00 1E00 2000 2200 2400 2600 2800 2A00 2C00 2E00 3000 3200 3400 3600 3800 3A00 3C00 3E00

5000 9000 D000

パターン ネーム テーブル

192 LINE

6000 A000 E000

212 LINE

6A00

7000 B000 F000

スプライト カラー テーブル

パレット テーブル

スプライト アトリビュート テーブル

スプライト ジェネレータ テーブル

パターンネームテーブル	0000H〜69FFH	27136	bytes
スプライトカラーテーブル	7400H〜75FFH	512	bytes
スプライトアトリビュートテーブル	7600H〜767FH	128	bytes
パレットテーブル	7680H〜769FH	32	bytes
スプライトジェネレータテーブル	7800H〜7FFFH	2048	bytes

図 A.7 SCREEN 5, 6 VRAM マップ

■SCREEN 7,8

GRAPHIC 6,7

パターンネームテーブル	0000H〜D3FFH	54272 bytes
スプライトジェネレータテーブル	F000H〜F7FFH	2048 bytes
スプライトカラーテーブル	F800H〜F9FFH	512 bytes
スプライトアトリビュートテーブル	FA00H〜FA7FH	128 bytes
パレットテーブル	FA80H〜FA9FH	32 bytes

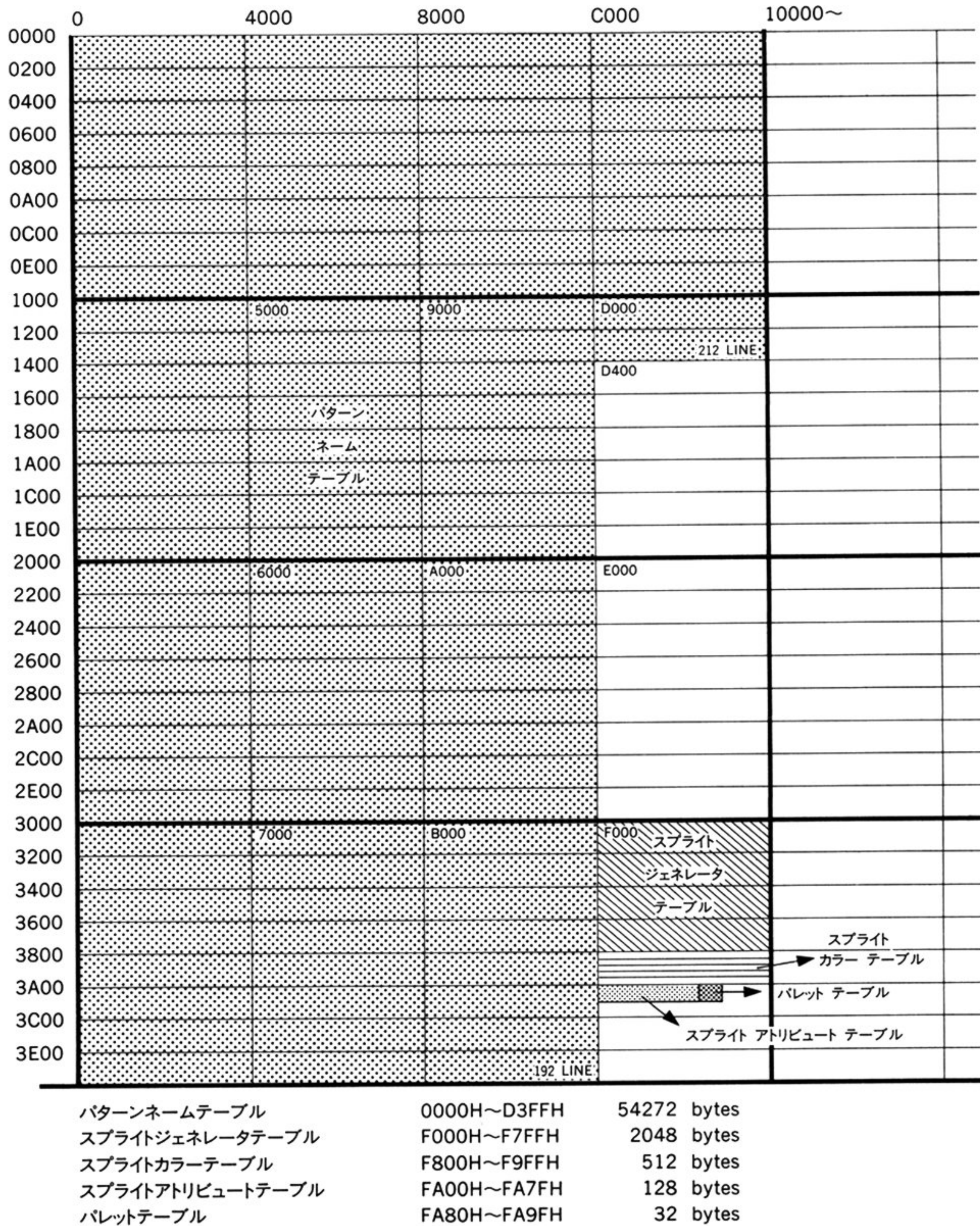



図 A.8 SCREEN 7, 8 VRAM マップ

表 A.1　VRAM アドレスの計算法

SCREEN	式
5	VRAM のページ番号×8000H＋先頭アドレス
6	VRAM のページ番号×8000H＋先頭アドレス
7	VRAM のページ番号×10000H＋先頭アドレス
8	VRAM のページ番号×10000H＋先頭アドレス

A.4 I/Oマップ

アドレス		内　容
00H～3FH		ユーザー定義
40H～4FH		拡張I/O
50H～6FH		システム予約
70H～73H		MIDI サウルス (BIT2)
74H～7BH		システム予約
7CH～7DH		MSX-MUSIC
7EH～7FH		システム予約
80H～87H		RS-232C 用
	80H	8251 データ
	81H	8251 ステータス・コマンド
	82H	ステータスリード・インタラプトマスク
	83H	未使用
	84H	8253
	85H	8253
	86H	8253
	87H	8253
88H～8BH		MSX2 アダプタのための VDP(V9938)用 I/O ポート MSX1 に V9938 をのせたときの I/O。VDP を直接 I/O アクセスするときはメイン ROM の 6 番地と 7 番地を調べてポートアドレスを確認する。
8CH～8DH		MSX MODEM
8EH～8FH		システム予約
90H～91H		プリンタポート
	90H	ビット 0 ストローブ出力 (Write)
	91H	ビット 1 ステータス入力 (Read)
92H～97H		システム予約
98H～9BH		VDP (V9938) MSX2 用
	98H	VRAM アクセス
	99H	コマンドレジスタアクセス
	9AH	パレットレジスタアクセス (Write のみ)
	9BH	レジスタ間接指定 (Write のみ)
9CH～9FH		システム予約
A0H～A3H		サウンドジェネレータ(AY-3-8910)
	A0H	アドレスラッチ
	A1H	データライト
	A2H	データリード
A4H～A7H		システム予約
A8H～ABH		パラレルポート (8255)
	A8H	ポート A

アドレス	内　容
A9H	ポートB
AAH	ポートC
ABH	モードセット
ACH～AFH	MSX ENGINE（1チップMSX I/O）
B0H～B3H	拡張メモリ（ソニー仕様、8255）
B0H	ポートA、アドレス(A0～A7)
B1H	ポートB、アドレス(A8～A10、A13～A15)、コントロール、Read/Write
B2H	ポートC、アドレス(A11～A12)、データ(D0～D7)
B3H	モードセット
B4H～B5H	CLOCK-IC(RP-5C01)
B4H	アドレスラッチ
B5H	データ
B6H～B7H	システム予約
B8H～BBH	ライトペンコントロール（三洋電機仕様）
B8H	Read/Write
B9H	Read/Write
BAH	Read/Write
BBH	Writeのみ
BCH～BFH	VHDコントロール（JVC、8255）
BCH	ポートA
BDH	ポートB
BEH	ポートC
BFH	モードセット
C0H～C1H	MSX-AUDIO
C2H～C7H	システム予約
C8H～CFH	MSX INTERFACE（非同期シリアル通信インターフェイス）
D0H～D7H	フロッピーディスクコントローラ(FDC)
	FDCは外部信号によって遮断される機能をもつ仕様になっており、このポートはFDCアクセス時に限って使用可能となる。
D8H～D9H	漢字ROM（東芝仕様）第1水準
D8H	b5～b0　下位アドレス（Writeのみ）
D9H	b5～b0　上位アドレス（Write）
	b7～b0　データ（Read）
DAH～DBH	漢字ROM第2水準
DCH～F2H	システム予約
F3H	VDPの表示モード(MSX2＋)
b0	M3
b1	M4
b2	M5
b3	M2
b4	M1
b5	TP

アドレス		内　容		
	b6	YUV		
	b7	YAE		
F4H	MSB	ハードウェアリセット（MSX2＋）		
F5H		システムコントロール（Writeのみ）bit ONでI/Oデバイス使用可		
	b0	漢字ROM第1水準		
	b1	漢字ROM第2水準		
	b2	MSX-AUDIO		
	b3	スーパーインポーズ		
	b4	MSX INTERFACE		
	b5	RS-232C		
	b6	ライトペン		
	b7	CLOCK-IC(MSX2、MSX2＋のみ実装)		
		内蔵あるいはカートリッジにより接続されたI/Oデバイスの衝突を避けるためのビット。これによって内蔵のデバイスを無効にすることができる。BIOSの初期化時に、外部デバイスがなければ内部デバイスを有効にする。ここはアプリケーションが読み書きしてはならない。		
F6H		カラーバスI/O		
F7H		A/Vコントロール		
	b0	オーディオR	ミキシング ON	(Write)
	b1	オーディオL	ミキシング OFF	(Write)
	b2	ビデオ入力の選択	21ピンRGB	(Write)
	b3	ビデオ入力の検出	入力なし	(Read)
	b4	AVコントロール	TV	(Write)
	b5	Ymコントロール	TV	(Write)
	b6	VDPレジスタ9のビット4の反転		(Write)
	b7	VDPレジスタ9のビット5の反転		(Write)
F8H～FBH		システム予約		
FCH～FFH		メモリマッパー（Writeのみ）		
	FCH	ページ3のセグメント選択レジスタ		
	FDH	ページ2のセグメント選択レジスタ		
	FEH	ページ1のセグメント選択レジスタ		
	FFH	ページ0のセグメント選択レジスタ		

A.5 拡張I/Oポート

A.5.1 概要

従来の MSX の仕様では、I/O ポートの 00H〜3FH までをユーザー領域、40H〜FFH を周辺機器用として割り当てていました。しかし、将来I/Oポートを希望する周辺機器すべてに別々の番地を割り当てることが不可能になると予想されます。

そこで、40H〜4FH までのI/O アドレスに割り当てられたデバイスは、選択されたときだけアクセスできるような構造にして、この領域を複数のデバイスが共用できるようにします。

40H 番地にデバイスの種類を表す特定のデータが書き込まれているときのみ、そのデバイスが 40H〜4FH のI/O ポートを使います。40H 番地に別のデータが書き込まれると、そのデバイスはI/O ポートの使用を中止して、バスバッファを CPU から切り離します(データバスからのリード動作は禁止しなくてもかまいません)。

A.5.2 ハードウェア例

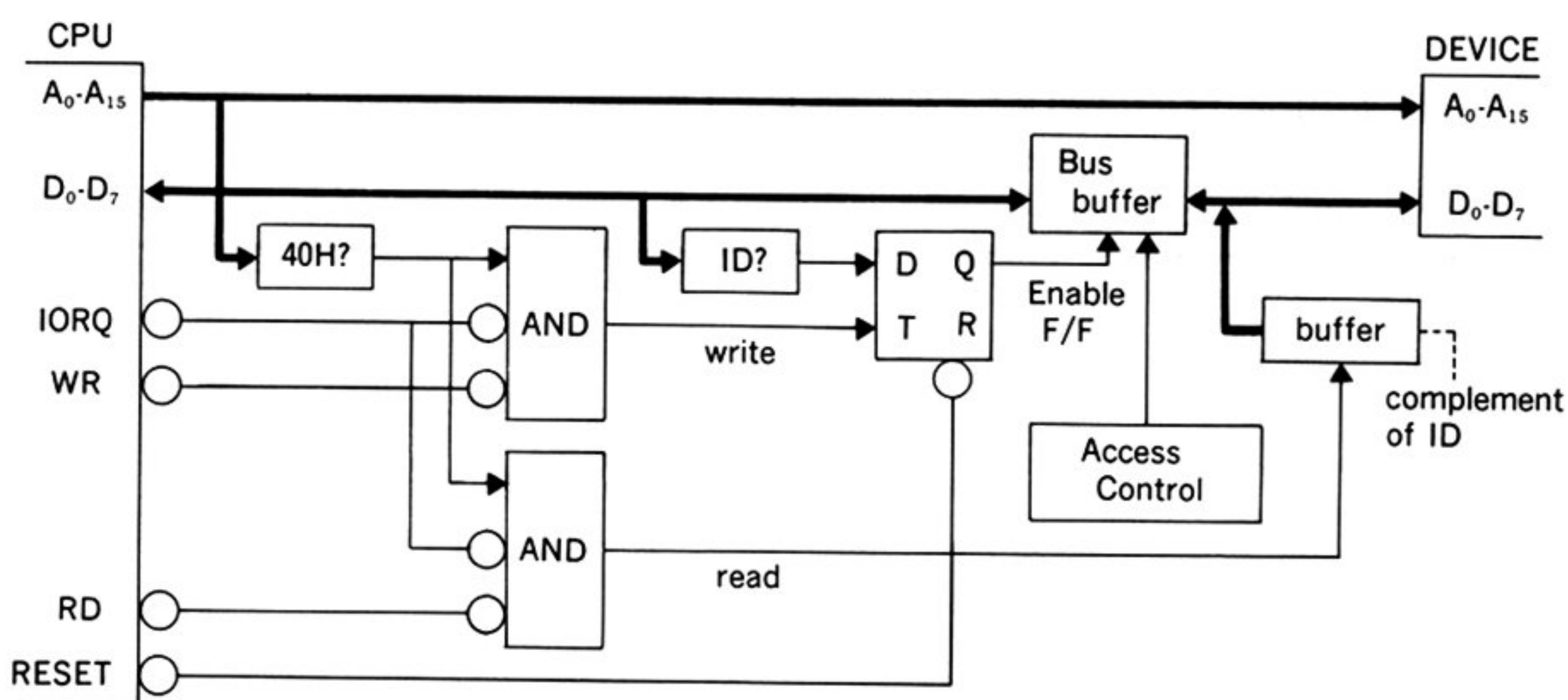



図 A.9　拡張I/Oポートに接続するデバイス

I/Oポートの40H番地に書き込まれた値が、そのデバイスのデバイスIDと一致すれば、CPUのバスにデバイスを接続します。デバイスIDが異なれば、CPUのバスからデバイスを切り離します。初期状態では、デバイスは切り離されています。

すでにデバイスが有効になっているときに、CPUが40H番地を読むと、デバイスIDの反転したデータを返します。これは、例えば割り込みで動作するプログラムが使用中のデバイスを知るために必要です。

A.5.3 デバイスID

デバイスIDには、0～255までの数値を割り当てることができます。しかし、デバイスIDの反転を読んで、デバイスの有無を検出するときの都合で、0と255は使用しません。1～127までは、拡張BIOSコールと同様にメーカーIDとして割り当てます。128から254までは、デバイスの種類に応じた番号を割り当てます。

原則として、特定の機種だけで使われる内蔵デバイスにはメーカーIDを、複数の機種に接続可能なデバイスにはデバイスの種類の番号を使います。

また、Z80CPUはI/O空間に16ビットアドレスを持っているので、将来さらに拡張が考えられるIDでは、上位8ビットもデコードして、16ビットでアクセスすることを推奨します。特に、メーカーIDで接続される装置は、16ビットでアクセスすることで、各IDごとのアドレス空間を256倍にすることができ、将来の拡張に耐えることになります。

表A.2　拡張I/OポートのメーカーID

メーカーID	メーカー名
1	アスキー
2	キヤノン
3	カシオ計算機
4	富士通
5	富士通ゼネラル
6	日立製作所
7	京セラ
8	松下電器産業
9	三菱電機
10	日本電気
11	ヤマハ
12	日本ビクター
13	フィリップス

メーカーID	メーカー名
14	パイオニア
15	三洋電機
16	シャープ
17	ソニー
18	スペクトラビデオ
19	東芝
20	ミツミ電機
21	テレマティカ
22	グラディエンテ
23	シャープドブラジル
24	GOLD STAR
25	DEAWOO
26	SumSong
27 ～ 127	予約

A.6 スーパーインポーズ

A.6.1 ハードウェア

1. I/O ポート F5H 番地のビット 3

内部 I/O ポート F7H 番地の入力を許可します。このビットが 1 のとき、本体内部の I/O ポート F7H 番地のリード（ビット 3）を有効にします（A.6.1 3.外部ポートの存在の判定」参照）。

この機能は I/O ポートのデータバスの衝突を避けるためのもので、他の方法により衝突が避けられる場合には、必ずしもこの機能を使う必要はありません。

2. I/O ポート F7H 番地

7	6	5	4	3	2	1	0

- 0：外部音声混合（右チャンネル） ［出力］
- 1：外部音声混合（左チャンネル） ［出力］
- 2：ビデオ入力選択 ［出力］
- 3：同期モード切り換え ［出力］
- 3：外部ビデオ検出 ［入力］
- 4：AV コントロール ［出力］
- 5：Ym コントロール ［出力］
- 7, 6：表 A.3 参照（VDP レジスタ# 9 のビット 4、5 に対応）

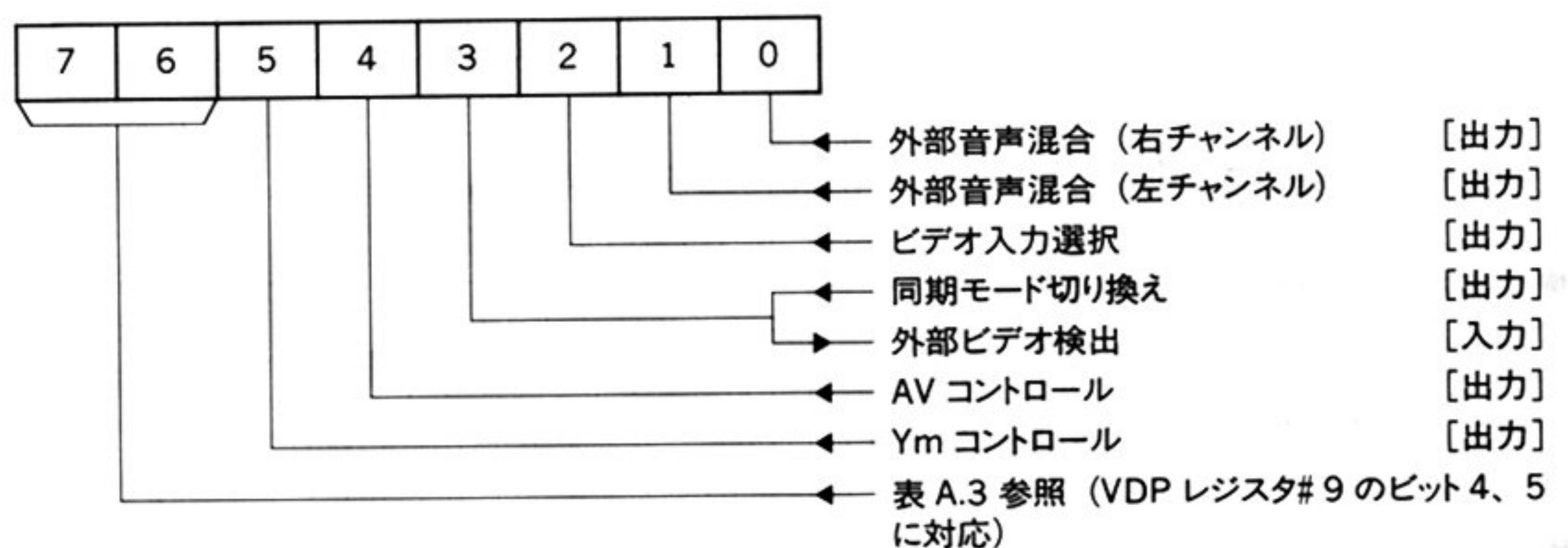



図 A.10　I/O ポート F7H のビット構成

■各ビットの詳細

ビット 7 とビット 6 には、VDP レジスタ#9 のビット 5（S1）、ビット 4（S0）の否定値が出力されます。したがって、ビット 7、6 の値と表示モードとの対応は、表 A.3 のようになります。

表 A.3　F7H のビット 7、6 と表示モードの対応

ビット 7	ビット 6	表示モード
0	0	未定義
0	1	テレビ画面（同期モードは NTSC コンパチブル。Ys が常時アクティブ）
1	0	コンピュータ画面・スーパーインポーズ*（同期モードは NTSC コンパチブル）
1	1	コンピュータ画面（同期モードは TMS9918 コンパチブル）

*コンピュータ画面とスーパーインポーズの切り換えは TP ビットで行う。TP ビット（VDP レジスタ#8 のビット 5）を 1 にすると、透明色の時に Ys 信号がアクティブになる。

● Ym コントロール（ビット 5)、AV コントロール（ビット 4)

マルチコネクタの出力を制御します。「0」のときに EIAJ TTC-003 に規定する論理レベル 0 を、「1」のときに論理レベル 1 を出力します。

●外部ビデオ検出（ビット 3、入力時)

ビデオ入力の有無を検出します。ビット 2 で選択された入力端子に入力がないときに「0」を、入力があるときに「1」を返します。ビット 7、ビット 6 で設定されるモードには関係しません。

●同期モード切り換え（ビット 3、出力時)

「0」のときに内部同期、「1」のときに外部同期になります。

●ビデオ入力選択（ビット 2)

ビデオおよびオーディオの入力端子を選択します。「0」のときに RGB マルチコネクタを、「1」のときに RCA コネクタを選択します。

●外部音声混合　左チャンネル（ビット 1)、右チャンネル（ビット 0)

外部オーディオ入力のミキシングを指定します。「0」のときにミキシング ON とし、コンピュータの音と外部入力の音を混合します。「1」のときにミキシング OFF とし、コンピュータの音のみを出力します。

3. 外部ポートの存在の判定

本体外部に接続する F7H ポートはリード時にビット 7 に「0」を返さなければなりません。外部に F7H ポートが接続されているときにこのポートを読むと、バスのプルアップ抵抗により「1」の入力が期待されます。

したがって、MSX2、MSX2＋のシステムソフトウェアは初期設定時に F7H ポートを読み、図

A.11 のビットパターンが返らなかったときに、外部 F7H ポートはないと判断し、F5H ポートのビット 3 を「1」にセットします。

7	6	5	4	3	2	1	0
0	1	1	1	×	1	1	1

図 A.11　初期設定時に判定される F7H ポートの値

4. 初期設定

MSX2、MSX2+のシステムソフトウェアにより、F7H 番地の初期設定が行われるので、ハードウェアによる初期設定は規定しません。

A.6.2 ソフトウェア

1. I/O ポート F7H 番地の初期設定

MSX2、MSX2+のシステムソフトウェアは、I/O ポートの F7H 番地を表 A.4 のように初期化します。

表 A.4　システムソフトウェアによる F7H の初期化

ビット	値	意　味
7	1	コンピュータ画面を表示する（TMS9918 コンパチブルモード）。
6	1	
5	0	フルトーンで表示する。
4	1	外部入力を選択する。
3	0	内部同期を選択する。
2	0	RGB マルチコネクタから入力する。
1、0	1	外部音声信号を混合しない。

これらのビットは、MSX BASIC version 2.0 以降でサポートされる、SET VIDEO ステートメントで操作することができます（「第 2 部 3 章 BASIC」参照）。

2. 各表示モードにおける標準設定値

各表示モードの標準設定値は表 A.5 のように定めます。

表 A.5　各モードの標準設定値

表示モード	7	6	5	4	3	2	1	0	(ビット)
コンピュータ	1	1	0	1	0	*1	1		
ビデオ　RGB マルチ	0	1	0	1	0	0	0		
ビデオ　RCA	0	1	0	1	0	1	0		
スーパーRGB マルチ	1	0	*2	1	1	0	*2		
スーパーRCA	1	0	*2	1	1	1	*2		

*1　任意

*2　「0」を基準とする。このモードに設定するときは、必要な場合にのみ対応するビットを「1」にしたパラメータをセットする。

各ビットをビット単位でセット・リセット可能にするために、ワークエリアの【AVCSAV(FAF7H)】に F7H ポートに出力した値を保存します。新たに F7H ポートの状態を変えるときは、このポートの内容の該当するビットを変更した後、【AVCSAC(FAF7H)】と F7H ポートに出力します。

3. BASIC の SET VIDEO 命令と I/O ポート F7H の関係

■ BASIC

SET VIDEO MODE, [Ym, [CB, [同期, [音声, [ビデオ入力, [AV コントロール]]]]]]

■ I/O ポート　F7H

7	6	5	4	3	2	1	0

表 A.6　MODE レジスタと S1、S0、CB の関係

MODE	S1	S0	意　味	HSYNC(HIGH)	HSYNC(LOW)	Ys
0	0	0	未定義			
1	0	1	コンピュータ画面	バーストフラグ	H カウントリセット入力	Low
2	0	1	スーパーインポーズ	HSYNC	H カウントリセット入力	High/Low
3	1	0	テレビ画面	HSYNC	H カウントリセット入力	High

表 A.7　V9938 の MODE レジスタ

レジスタ	7	6	5	4	3	2	1	0	
R#8	MS	LP	TP	CB	VR	0	SPD	BW	MODE レジスタ 2
R#9	LN	0	S1	S0	IL	EO	NT	DC	MODE レジスタ 3

注　意　ビット 7、6 は、それぞれ S1、S0 を反転したもの

表 A.8　Ym とビット 5

Ym	ビット 5	意　味
0	0	テレビの輝度　通常
1	1	テレビの輝度　半分

表 A.9　同期とビット 3

同期	ビット 3	意　味
0	0	内部同期
1	1	外部同期

表 A.10　音声とビット 1、0

音声	ビット 1 (L)	ビット 0 (R)	意　味
0	1	1	外部音声を混合しない
1	1	0	右チャンネルの外部音声信号と混合
2	0	1	左チャンネルの外部音声信号と混合
3	0	0	両チャンネルの外部音声信号と混合

表 A.11　ビデオ入力とビット 2

ビデオ入力	ビット 2	意　味
0	0	RGB マルチコネクタを選択
1	1	RCA コネクタを選択

表 A.12　AV コントロールとビット 4

AV コントロール	ビット 4	意　味
0	0	RGB マルチコネクタの AV コントロール端子の出力　OFF
1	1	RGB マルチコネクタの AV コントロール端子の出力　ON

A.6.3 その他

RGB マルチコネクタの AV コントロール、Ys、Ym 信号出力は、本体の電源が切れているときに、EIAJ TTC-003 に規定する論理レベル 0 でなければなりません。この条件を守らなければ、AV 対応テレビの映像が正しく表示されないことがあります。

A.7 サンプルプログラム

サンプルディスクに入っているプログラム(以下、「サンプルプログラム」とします)は、「MSX-Datapack」の登録ユーザーの方であれば、ユーザープログラムに組み込んで使用することができます。そのユーザープログラムを販売・頒布する場合も、弊社との契約は必要ありません。その際に、弊社の Copyright 表示などは必要ありません。

ただし、サンプルプログラムの著作権は株式会社アスキーが保有します。したがって、一部、全部に関わらず、サンプルプログラムの内容をそのまま、単体で第三者に販売・頒布することは禁止します。同様に、パソコン通信において、サンプルプログラムのソースプログラムまたはオブジェクトプログラムを単体でホストコンピュータにアップロードすることは禁止します。

サンプルプログラムの内容、マニュアルの正誤情報は、サンプルディスクの以下のファイルに記録していますのでご参照下さい。

ファイル名	内　容
README.DOC	マニュアルの正誤情報など
CONTENTS.DOC	サンプルディスクに含まれているファイルの内容

これらのファイルは漢字を含んだファイルです。したがって、漢字 ROM を搭載した MSX2、MSX2＋で以下のコマンドをキーボードから入力すると画面に表示することができます。

```
A>jt readme.doc
A>jt contents.doc
```

また、サンプルプログラムを運用した結果の影響については、弊社は責任を負いませんのでご了承下さい。

A.8 エスケープシーケンス

カーソル移動	
<ESC>A	カーソルを上に移動
<ESC>B	カーソルを下に移動
<ESC>C	カーソルを右に移動
<ESC>D	カーソルを左に移動
<ESC>H	カーソルをホームポジションに移動
<ESC>Y<Y座標+20H><X座標+20H>	カーソルを(X,Y)の位置に移動

編集、削除	
<ESC>j	画面をクリア
<ESC>E	画面をクリア
<ESC>K	行の終わりまで削除
<ESC>l	行の終わりまで削除
<ESC>J	画面の終わりまで削除
<ESC>L	1行挿入
<ESC>M	1行削除

その他	
<ESC>x4	カーソルの形を■にする
<ESC>x5	カーソルを消す
<ESC>y4	カーソルの形を _ にする
<ESC>y5	カーソルを表示する

A.9 コントロールコード

10進	16進	機　能	対応キー
0	00H		CTRL + @
1	01H	グラフィックキャラクタの出力時のヘッダ	CTRL + A
2	02H	カーソルを直前の語の先頭へ移動	CTRL + B
3	03H	入力待ち状態を終了	CTRL + C
4	04H		CTRL + D
5	05H	カーソル以降を削除	CTRL + E
6	06H	カーソルを次の語の先頭へ移動	CTRL + F
7	07H	スピーカーを鳴らす（BEEP 文と同じ）	CTRL + G
8	08H	カーソルの 1 つ前の文字を削除	CTRL + H、BS
9	09H	次の水平タブ位置へ移動	CTRL + I、TAB
10	0AH	行送り（ラインフィード）	CTRL + J
11	0BH	カーソルをホームポジション（左上）に戻す	CTRL + K、HOME
12	0CH	画面をクリアし、カーソルをホームポジションに戻す	CTRL + L、CLS
13	0DH	カーソルを左端に戻す（キャリッジリターン）	CTRL + M、RETURN
14	0EH	カーソルを行末へ移動	CTRL + N
15	0FH		CTRL + O
16	10H		CTRL + P
17	11H		CTRL + Q
18	12H	挿入モードの ON/OFF スイッチ	CTRL + R、INS
19	13H		CTRL + S
20	14H		CTRL + T
21	15H	1 行を画面から削除	CTRL + U
22	16H		CTRL + V
23	17H		CTRL + W
24	18H		CTRL + X、SELECT
25	19H		CTRL + Y
26	1AH		CTRL + Z
27	1BH		CTRL + [、ESC
28	1CH	カーソルを右へ移動	CTRL + ¥、→
29	1DH	カーソルを左へ移動	CTRL +]、←
30	1EH	カーソルを上へ移動	CTRL + ^、↑
31	1FH	カーソルを下へ移動	CTRL + _、↓
127	7FH	カーソルの指す文字を削除	DEL

A.10 キャラクタコード表

コード(10進)	コード(16進)	キャラクタ	コード(10進)	コード(16進)	キャラクタ	コード(10進)	コード(16進)	キャラクタ	コード(10進)	コード(16進)	キャラクタ
0	00H	↑	32	20H	空白	64	40H	@	96	60H	'
1	01H		33	21H	!	65	41H	A	97	61H	a
2	02H		34	22H	″	66	42H	B	98	62H	b
3	03H		35	23H	#	67	43H	C	99	63H	c
4	04H		36	24H	$	68	44H	D	100	64H	d
5	05H		37	25H	%	69	45H	E	101	65H	e
6	06H		38	26H	&	70	46H	F	102	66H	f
7	07H		39	27H	'	71	47H	G	103	67H	g
8	08H		40	28H	(	72	48H	H	104	68H	h
9	09H		41	29H	)	73	49H	I	105	69H	i
10	0AH		42	2AH	*	74	4AH	J	106	6AH	j
11	0BH	コ	43	2BH	+	75	4BH	K	107	6BH	k
12	0CH	ン	44	2CH	,	76	4CH	L	108	6CH	l
13	0DH	ト	45	2DH	−	77	4DH	M	109	6DH	m
14	0EH	ロ	46	2EH	.	78	4EH	N	110	6EH	n
15	0FH	ール	47	2FH	/	79	4FH	O	111	6FH	o
16	10H	キ	48	30H	0	80	50H	P	112	70H	p
17	11H	ャ	49	31H	1	81	51H	Q	113	71H	q
18	12H	ラ	50	32H	2	82	52H	R	114	72H	r
19	13H	ク	51	33H	3	83	53H	S	115	73H	s
20	14H	タ	52	34H	4	84	54H	T	116	74H	t
21	15H		53	35H	5	85	55H	U	117	75H	u
22	16H		54	36H	6	86	56H	V	118	76H	v
23	17H		55	37H	7	87	57H	W	119	77H	w
24	18H		56	38H	8	88	58H	X	120	78H	x
25	19H		57	39H	9	89	59H	Y	121	79H	y
26	1AH		58	3AH	:	90	5AH	Z	122	7AH	z
27	1BH		59	3BH	;	91	5BH	[	123	7BH	{
28	1CH		60	3CH	<	92	5CH	¥	124	7CH	|
29	1DH		61	3DH	=	93	5DH	]	125	7DH	}
30	1EH		62	3EH	>	94	5EH	^	126	7EH	~
31	1FH	↓	63	3FH	?	95	5FH	_	127	7FH	削除

コード (10進)	コード (16進)	キャラクタ	コード (10進)	コード (16進)	キャラクタ	コード (10進)	コード (16進)	キャラクタ	コード (10進)	コード (16進)	キャラクタ
128	80H	♠	160	A0H		192	C0H	ﾀ	224	E0H	た
129	81H	♥	161	A1H	｡	193	C1H	ﾁ	225	E1H	ち
130	82H	♣	162	A2H	｢	194	C2H	ﾂ	226	E2H	つ
131	83H	◆	163	A3H	｣	195	C3H	ﾃ	227	E3H	て
132	84H	○	164	A4H	､	196	C4H	ﾄ	228	E4H	と
133	85H	●	165	A5H	･	197	C5H	ﾅ	229	E5H	な
134	86H	を	166	A6H	ｦ	198	C6H	ﾆ	230	E6H	に
135	87H	ぁ	167	A7H	ｧ	199	C7H	ﾇ	231	E7H	ぬ
136	88H	ぃ	168	A8H	ｨ	200	C8H	ﾈ	232	E8H	ね
137	89H	ぅ	169	A9H	ｩ	201	C9H	ﾉ	233	E9H	の
138	8AH	ぇ	170	AAH	ｪ	202	CAH	ﾊ	234	EAH	は
139	8BH	ぉ	171	ABH	ｫ	203	CBH	ﾋ	235	EBH	ひ
130	8CH	ゃ	172	ACH	ｬ	204	CCH	ﾌ	236	ECH	ふ
141	8DH	ゅ	173	ADH	ｭ	205	CDH	ﾍ	237	EDH	へ
142	8EH	ょ	174	AEH	ｮ	206	CEH	ﾎ	238	EEH	ほ
143	8FH	っ	175	AFH	ｯ	207	CFH	ﾏ	239	EFH	ま
144	90H		176	B0H	ｰ	208	D0H	ﾐ	240	F0H	み
145	91H	あ	177	B1H	ｱ	209	D1H	ﾑ	241	F1H	む
146	92H	い	178	B2H	ｲ	210	D2H	ﾒ	242	F2H	め
147	93H	う	179	B3H	ｳ	211	D3H	ﾓ	243	F3H	も
148	94H	え	180	B4H	ｴ	212	D4H	ﾔ	244	F4H	や
149	95H	お	181	B5H	ｵ	213	D5H	ﾕ	245	F5H	ゆ
150	96H	か	182	B6H	ｶ	214	D6H	ﾖ	246	F6H	よ
151	97H	き	183	B7H	ｷ	215	D7H	ﾗ	247	F7H	ら
152	98H	く	184	B8H	ｸ	216	D8H	ﾘ	248	F8H	り
153	99H	け	185	B9H	ｹ	217	D9H	ﾙ	249	F9H	る
154	9AH	こ	186	BAH	ｺ	218	DAH	ﾚ	250	FAH	れ
155	9BH	さ	187	BBH	ｻ	219	DBH	ﾛ	251	FBH	ろ
156	9CH	し	188	BCH	ｼ	220	DCH	ﾜ	252	FCH	わ
157	9DH	す	189	BDH	ｽ	221	DDH	ﾝ	253	FDH	ん
158	9EH	せ	190	BEH	ｾ	222	DEH	ﾞ	254	FEH	
159	9FH	そ	191	BFH	ｿ	223	DFH	ﾟ	255	FFH	カーソル表示

A.11 グラフィックキャラクタコード表

グラフィックキャラクタはすべて2バイト文字です。CHR$(1)を識別用のヘッダとして、続けて以下のコードを与えることによって表現します。

コード(10進)	コード(16進)	キャラクタ	コード(10進)	コード(16進)	キャラクタ
64	40H		80	50H	π
65	41H	月	81	51H	┴
66	42H	火	82	52H	┬
67	43H	水	83	53H	┤
68	44H	木	84	54H	├
69	45H	金	85	55H	┼
70	46H	土	86	56H	│
71	47H	日	87	57H	─
72	48H	年	88	58H	┌
73	49H	円	89	59H	┐
74	4AH	時	90	5AH	└
75	4BH	分	91	5BH	┘
76	4CH	秒	92	5CH	□
77	4DH	百	93	5DH	大
78	4EH	千	94	5EH	中
79	4FH	万	95	5FH	小

A.12 トークン順の中間コード一覧

81	END	AC	DEFINT	D7	CMD	FF86	ABS
82	FOR	AD	DEFSNG	D8	LOCATE	FF87	SQR
83	NEXT	AE	DEFDBL	D9	TO	FF88	RND
84	DATA	AF	LINE	DA	THEN	FF89	SIN
85	INPUT	B0	OPEN	DB	TAB(	FF8A	LOG
86	DIM	B1	FIELD	DC	STEP	FF8B	EXP
87	READ	B2	GET	DD	USR	FF8C	COS
88	LET	B3	PUT	DE	FN	FF8D	TAN
89	GOTO	B4	CLOSE	DF	SPC(	FF8E	ATN
8A	RUN	B5	LOAD	E0	NOT	FF8F	FRE
8B	IF	B6	MERGE	E1	ERL	FF90	INP
8C	RESTORE	B7	FILES	E2	ERR	FF91	POS
8D	GOSUB	B8	LSET	E3	STRING$	FF92	LEN
8E	RETURN	B9	RSET	E4	USING	FF93	STR$
8F	REM	BA	SAVE	E5	INSTR	FF94	VAL
90	STOP	BB	LFILES	3A8FE6	'	FF95	ASC
91	PRINT	BC	CIRCLE	E7	VARPTR	FF96	CHR$
92	CLEAR	BD	COLOR	E8	CSRLIN	FF97	PEEK
93	LIST	BE	DRAW	E9	ATTR$	FF98	VPEEK
94	NEW	BF	PAINT	EA	DSKI$	FF99	SPACE$
95	ON	C0	BEEP	EB	OFF	FF9A	OCT$
96	WAIT	C1	PLAY	EC	INKEY$	FF9B	HEX$
97	DEF	C2	PSET	ED	POINT	FF9C	LPOS
98	POKE	C3	PRESET	EE	>	FF9D	BIN$
99	CONT	C4	SOUND	EF	=	FF9E	CINT
9A	CSAVE	C5	SCREEN	F0	<	FF9F	CSNG
9B	CLOAD	C6	VPOKE	F1	+	FFA0	CDBL
9C	OUT	C7	SPRITE	F2	−	FFA1	FIX
9D	LPRINT	C8	VDP	F3	*	FFA2	STICK
9E	LLIST	C9	BASE	F4	/	FFA3	STRIG
9F	CLS	CA	CALL	F5	^	FFA4	PDL
A0	WIDTH	CB	TIME	F6	AND	FFA5	PAD
3AA1	ELSE	CC	KEY	F7	OR	FFA6	DSKF
A2	TRON	CD	MAX	F8	XOR	FFA7	FPOS
A3	TROFF	CE	MOTOR	F9	EQV	FFA8	CVI
A4	SWAP	CF	BLOAD	FA	IMP	FFA9	CVS
A5	ERASE	D0	BSAVE	FB	MOD	FFAA	CVD
A6	ERROR	D1	DSKO$	FC	¥	FFAB	EOF
A7	RESUME	D2	SET	FF81	LEFT$	FFAC	LOC
A8	DELETE	D3	NAME	FF82	RIGHT$	FFAD	LOF
A9	AUTO	D4	KILL	FF83	MID$	FFAE	MKI$
AA	RENUM	D5	IPL	FF84	SGN	FFAF	MKS$
AB	DEFSTR	D6	COPY	FF85	INT	FFB0	MKD$

A13. ファンクションコール一覧

番号	機能	
00H	**システムリセット**	
	コール手順	戻り値
	なし	なし
01H	**コンソール入力（入力待ち、エコーバック、CTRL コードチェックあり）**	
	コール手順	戻り値
	なし	A＝入力した文字
02H	**コンソール出力**	
	コール手順	戻り値
	E＝出力する文字	なし
03H	**補助入力**	
	コール手順	戻り値
	なし	A＝AUX から読み込んだ文字
04H	**補助出力**	
	コール手順	戻り値
	E＝AUX に出力する文字	なし
05H	**プリンタ出力**	
	コール手順	戻り値
	E＝プリンタに出力する文字	
06H	**直接コンソール入出力（入力待ち、エコーバック、CTRL コードチェックなし）**	
	コール手順	戻り値
	E＝0FFH なら入力、それ以外は出力	A＝文字またはヌル
07H	**直接コンソール入力（入力待ちあり、エコーバック、CTRL コードチェックなし）**	
	コール手順	戻り値
	なし	A＝文字コード
08H	**コンソール入力（入力待ちあり、エコーバックなし、CTRL コードチェックあり）**	
	コール手順	戻り値
	なし	A＝文字コード
09H	**文字列出力**	
	コール手順	戻り値
	DE＝文字列の先頭アドレス	なし
0AH	**文字列入力**	
	コール手順	戻り値
	DE＝最大文字数をセットしたアドレス	(DE＋1～)＝文字数と文字列
0BH	**コンソール入力状態のチェック**	
	コール手順	戻り値
	なし	A　0FFH＝入力あり、00H＝なし

番号	機　能	
0CH	**バージョン番号の獲得** コール手順 なし	戻り値 HL＝0022H
0DH	**ディスクリセット** コール手順 なし	戻り値 なし
0EH	**デフォルトドライブの設定** コール手順 E＝デフォルトドライブ番号(A＝0…)	戻り値 なし
0FH	**ファイルのオープン** コール手順 DE ＝オープンされてないFCBの先頭アドレス FCBのカレントブロック ＝読み出すブロック番号	戻り値 A　00H＝オープン成功、0FFH＝失敗
10H	**ファイルのクローズ** コール手順 DE＝オープンされたFCBの先頭アドレス	戻り値 A　00H＝クローズ成功、0FFH＝失敗
11H	**ファイルの検索（最初の一致）** コール手順 DE ＝オープンされてないFCBの先頭アドレス	戻り値 A　00H＝発見、0FFH＝発見できず
12H	**ファイルの検索（後続の一致）** コール手順 なし	戻り値 A　00H＝発見、0FFH＝発見できず
13H	**ファイルの削除** コール手順 DE ＝オープンされてないFCBの先頭アドレス	戻り値 A　00H＝削除成功、0FFH＝失敗
14H	**シーケンシャルな読み出し** コール手順 DE＝オープンされたFCBの先頭アドレス FCBのカレントレコード ＝読み出すレコード番号	戻り値 A　00H＝読み出し成功、0FFH＝失敗 DTA 領域＝読み込んだレコード
15H	**シーケンシャルな書き込み** コール手順 DE＝オープンされたFCBの先頭アドレス FCBのカレントレコード ＝書き込むレコード番号 DTA 以降の128バイト＝書き込むデータ	戻り値 A　00H＝書き込み成功、0FFH＝失敗
16H	**ファイルの作成** コール手順 DE ＝オープンされてないFCBの先頭アドレス FCBのカレントレコード ＝読み出すブロック番号	戻り値 A　00H＝作成成功、0FFH＝失敗

番号	機　能	
17H	**ファイル名の変更**	
	コール手順 DE =オープンされてないFCBの先頭アドレス FCB+0=旧ファイル名 FCB+16=新ファイル名	戻り値 A　00H=変更成功、0FFH=失敗
18H	**ログインベクトルの獲得**	
	コール手順 なし	戻り値 HL=オンラインドライブ情報 LのLSBがA：、MSBがH：に対応
19H	**デフォルトドライブ番号の獲得**	
	コール手順 なし	戻り値 A=デフォルトドライブ番号(A=0…)
1AH	**転送先アドレス（DTA）の設定**	
	コール手順 DE=転送するDTAアドレス	戻り値 なし
1BH	**ディスク情報の獲得**	
	コール手順 E=目的ディスクのドライブ番号	戻り値 A=1クラスタあたりの論理セクタ数 =FFHなら指定ドライブがオフライン BC=セクタのサイズ(バイト単位) DE=クラスタの総数 HL=未使用クラスタの総数 IX=DPBの先頭アドレス IY=FATバッファ先頭アドレス
1CH〜20H	**無効**	
21H	**ランダムな読み出し**	
	コール手順 DE=オープンされたFCBの先頭アドレス FCBランダムレコード =読み出すレコード番号	戻り値 A　00H=読み出し成功、01H=失敗 DTA 領域=読み出したレコード
22H	**ランダムな書き込み**	
	コール手順 DE=オープンされたFCBの先頭アドレス FCBランダムレコード =書き込むレコード番号 DTA以降の128バイト=書き込む内容	戻り値 A　00H=書き込み成功、01H=失敗
23H	**ファイルサイズの獲得**	
	コール手順 DE=オープンされたFCBの先頭アドレス	戻り値 A　00H=獲得成功、0FFH=失敗 FCBのランダムレコードフィールド =ファイルサイズ(128バイト単位)

番号	機　能	
24H	**ランダムレコードフィールドの設定**	
	コール手順 DE=オープンされたFCBの先頭アドレス FCBカレントレコードブロック =目的のブロック FCBカレントレコード=目的のレコード	戻り値 FCBのランダムレコードフィールド =カレントレコードの位置
25H	**無効**	
26H	**ランダムブロック書き込み**	
	コール手順 DE=オープンされたFCBの先頭アドレス HL=書き込むレコード数 FCBのレコードサイズ=書き込むサイズ FCBのランダムレコード=開始レコード DTA以降のメモリ=書き込むデータ	戻り値 A　00H=書き込み成功、01H=失敗
27H	**ランダムブロック読み出し**	
	コール手順 DE=オープンされたFCBの先頭アドレス HL=読み出すレコード数 FCBのレコードサイズ=読み出すサイズ FCBのランダムレコード=開始レコード	戻り値 A　00H=書き込み成功、01H=失敗 HL=実際に読んだレコード数
28H	**「0」の書き込みをともなうランダムな書き込み**	
	コール手順 DE=オープンされたFCBの先頭アドレス FCBランダムレコード =書き込むレコード番号 DTA以降の128バイト=書き込む内容	戻り値 A　00H=書き込む成功、01H=失敗
29H	**無効**	
2AH	**日付の獲得**	
	コール手順 なし	戻り値 HL=年(1980～2079) D=月(1～12) E=日(1～31) A=曜日(0=日曜…)
2BH	**日付の設定**	
	コール手順 HL=年(1980～2079) D=月(1～12) E=(1～31)	戻り値 A　00H=設定成功、0FFH=失敗
2CH	**時刻の獲得**	
	コール手順 なし	戻り値 H=時 L=分 D=秒 E=1/100秒

番号	機　能	
2DH	**時刻の設定**	
	コール手順 H＝時 L＝分 D＝秒	戻り値 A　00H＝設定成功、0FFH＝失敗
2EH	**ベリファイフラグの設定**	
	コール手順 E　00H＝リセット、00H以外＝セット	戻り値 なし
2FH	**論理セクタの読み出し**	
	コール手順 DE＝読み出すセクタ番号 H＝読み出すセクタの個数 L＝ディスクのドライブ番号(A＝0…)	戻り値 DTAバッファ＝読んだ内容
30H	**論理セクタの書き込み**	
	コール手順 DE＝書き込むセクタ番号 H＝書き込むセクタの個数 L＝ディスクのドライブ番号(A＝0…) DTAアドレスのメモリ＝書き込む内容	戻り値 なし

索引

(<　>内は Volume 1 のページ数を示します)

参考文献

「MSX テクニカルデータブック 1　増補改訂版」　アスキー

「MSX2 テクニカルハンドブック」　アスキー

「V9938 MSX-VIDEO テクニカルデータブック」　アスキー

「YM2413 アプリケーションマニュアル」　ヤマハ

「Y8950 アプリケーションマニュアル」　ヤマハ

お問い合わせについて

弊社では厳重に梱包した上、細心の注意を払って製品を発送しております。万一、輸送上のトラブルが起こった場合には、ご一報いただければ新しいものと交換いたします。

マニュアルの作成については、なるべく詳細な説明をするよう心がけたつもりですが、理解できないところは、実際にコンピュータと向き合って納得のいくまで確かめて下さい。また、他のページを参照するのもひとつの方法です。それでも疑問点が解決できないときは、封書にて、下記の要領でお問い合わせ下さい。恐れ入りますが、このパッケージの性格上、お電話でのお答えは不可能と存じますので、ご了承下さいますようお願い申し上げます。

記

1. 送付先　〒107-24 東京都港区南青山6-11-1　スリーエフ南青山ビル
　株式会社アスキー　ユーザーサポート係
2. 必要項目
   (1) お客様の氏名、郵便番号、住所、電話番号（市外局番も含む）
   (2) 製品名、ユーザーID番号、製品シリアル番号
   (3) 機器構成
      本体のメーカー名、機種名
      接続している周辺機器のメーカー名、機種名（型番）
      使用しているソフトウェアのメーカー名とその名前
3. お問い合わせ内容

お問い合わせの内容は、製品のマニュアルに記述されている用語を用いて、具体的かつ明確に記述して下さい。なお、障害と思われる現象については、その現象を再現可能な情報が必要です。当社で再現できないものは調査できません。その現象が発生するまでの操作手順、データを必ず添付して下さい。データディスクがある場合は、そのコピーも同封して下さい。

なお、お客様がプログラミングされたアプリケーションソフトウェアの設計、作成、運用、保守などについては、当社のサポート範囲外ですので、お問い合わせいただいても回答できません。例えば、「このプログラムリストはなぜ動かないのか」といった質問にはお答えできません。

また、MSX-MUSICとMSX-AUDIO用のFM音源LSIの技術資料は、ヤマハ株式会社のご好意により掲載させていただきましたが、この内容についての問い合わせも弊社で承ります。ただし、具体的な音作りなどに関するご質問にはお答えできませんので、ご了承下さいますようお願い申し上げます。

MSX-Datapack Volume 2

1991年2月1日　第1版第1刷

編　　集　株式会社アスキー　システム事業部

発 行 所　株式会社アスキー

担当　遠藤　祥、北浦　訓行

〒107-24 東京都港区南青山6-11-1 スリーエフ南青山ビル

TEL (03)3486-7111(大代表)

制　　作　株式会社ジャパックスインターナショナル